Terraform StateFile

Consider it as the reference for the reality of the existing infrastructure which you can refer to, from inside your code.

So basically, a state file is the reality of your deployment on your cloud provider from the intention declared on your Terraform code.

Terraform StateFile storing in s3 and best managing practices

The first time you initialize your git repo with Terraform code and apply it is going to create the TFState directly on the root of your github repo but having that in github repo leads to exposing the credentials and secure data. So I use the s3 bucket by asking terraform backend and provide the s3 bucket details and assign bucket policy such that only the ec2 instance arn can use this file. S3 supports server-side encryption using AES-256

Terraform uses SSL to read and write data in S3

**Locking**: Most version control systems do not provide any form of locking that would prevent two team members from running terraform apply on the same state file at the same time.

terraform workspace new example1

Terraform lost statefiles  
  
In order to regain control of this resource we will use the import command to recover the metadata for this resource. Import requires a resource address and the resource ID. The resource address is a combination of the resource type and resource name which for this configuration is ‘aws\_instance.example’. The ID in this instance is the AWS EC2 ID ‘i-06becd0903c31ab3e’.

terraform refresh

is used to know the current infrastructure with terraform state and detect if there is any drift.

terraform workspace new example1

The default workspace stores your state in exactly the location you specify via the key configuration. If you take a look in your S3 bucket, you’ll find a terraform.tfstate file in the workspaces-example folder:

Setting Up a Jenkins Continuous Deployment

Next, we will look into setting up a Jenkins server which will be used for our CI/CD pipeline. We will be using Terraform and AWS for setting this up as well.

The Terraform code for setting Jenkins is inside the folder jenkins/setup. Let’s go through some of the interesting things about this setup.

1. AWS credentials: You can either provide the AWS access key ID and secret access key to the Terraform AWS provider (instance.tf) or you can give the location of credentials file to the property shared\_credentials\_file in the AWS provider.
2. IAM role: Since we will be running Packer and Terraform from the Jenkins server, they would be accessing S3, EC2, RDS, IAM, load balancing, and autoscaling services on AWS. So either we provide our credentials on Jenkins for Packer & Terraform to access these services or we can create an IAM Profile (iam.tf), using which we would create a Jenkins instance.
3. Terraform state: Terraform has to maintain the state of the infrastructure somewhere in a file and, with S3 (backend.tf), you could just maintain it there, so you can collaborate with other coworkers, and anyone can change and deploy since the state is maintained in a remote location.
4. Public/private key pair: You will need to upload the public key of your key pair along with the instance so that you can ssh into the Jenkins instance once it is up. We have defined an aws\_key\_pair resource (key.tf) in which you specify the location of your public key using Terraform variables.

Steps for setting up Jenkins:

Step 1: For keeping the remote state of Terraform, you would need to manually create a bucket in S3 which can be used by Terraform. This would be the only step done outside of Terraform. Make sure you run AWS configure before running the command below to specify your AWS credentials.

aws s3api create-bucket --bucket node-aws-jenkins-terraform --region eu-west-1 --create-bucket-configuration LocationConstraint=eu-west-1

Step 2: Run terraform init. This will initialize the state and configure it to be stored on S3 and download the AWS provider plugin.

Step 3: Run terraform apply. This will check all the Terraform code and create a plan and show how many resources will be created after this step has finished.

Step 4: Type yes, and then the previous step will start creating all the resources. After the command finishes, you will get the public IP address of the Jenkins server.

Step 5: Ssh into the Jenkins server, using your private key. ubuntu is the default username for AWS EBS-backed instances. Use the IP address returned by the terraform apply command.

ssh -i mykey ubuntu@34.245.4.73

Step 6: Start the Jenkins web UI by going to http://34.245.4.73:8080. The password can be found at /var/lib/jenkins/secrets/initialAdminPassword.

Step 7: Choose “Install Suggested Plugins” and Create an user for Jenkins.

Ansible Questions(2.9)

Ansible is broken down into two types of servers: **controlling machines and nodes**. Ansible is installed on the controlling computer, and the controlling machines manage the nodes via SSH.

The controlling machine contains an **inventory file** that holds the node system’s location. Ansible runs the playbook on the controlling machine to deploy the modules on the node systems. Since Ansible is agentless, there’s no need for a third-party tool to connect the nodes.

**Playbook**

playbook has a series of YAML-based files that send commands to remote computers via scripts

• Task: this is a call to a particular Ansible module

• Module: this is an actual unit of code, which is implemented by Ansible on your host or a remote type of host. The modules can be indexed by category, which is also denoted as the task plugins.

• Play: One or more of the tasks executed on a particular host.

• Playbook: One or more of the plays. Each of the plays may be executed on similar or different hosts.

• Role: Ansible roles allow you to group the resources according to particular functionality or service such that they can be used easily. In this role, you have directories for variables, templates, handlers, tasks files and metadata. It is possible to then use the role through specifying it within the playbook.

**Ansible Tower**

Tower’s primary function is to serve as the hub for all of an organization’s automation tasks, allowing users to monitor configurations and conduct rapid deployments

Scale Capacity With Clusters  
Schedule Ansible Jobs  
Integrated Notifications

**generate encrypted passwords for the User Module**

The mkpasswd utility which is available on a lot of the Linux systems is one of the great options whereby mkpasswd—method= 512. If the utility is not installed on the system then you may still generate the passwords through Python. First, make sure the Passlib password-hashing library has been installed.

How do you get **Ansible to reuse the connection** and allow **kerberized SSH** or have the platform pay attention to your local SSH config file?

Switch the default connect type within the configuration file to SSH or use ‘-C SSH’ in order to use Native OpenSSH for the connections as opposed to the python paramiko library. In Ansible 1.2.1 and then SSH is going to be utilized by default if the OpenSSH type provides a number of advanced alternatives. So you will want to run Ansible from a machine that is new such that it can support ControlPersist, in the event that you are using this connection type. You will still be able to manage some of the older clients. In the event you are using RHEL 6, CentOS 6, SLES 10 or even SLES 11, then the version of OpenSSH is still old so it would be better to consider managing from a Fedora or openSUSE client even though you may be managing older nodes or you can just use paramiko. Paramiko is kept as the default as you may be first installing Ansible on an EL box so it provides one of the better experiences for the new users.

**What is the difference between environment variables and variable name**

A variable name may be built by adding String but an environment variable can be accessed by accessing the existing variable.

• For the Environment Variable, if you want to add the variable, you need to open the advance playbooks section. For the Variable Name, you can add Strings. • For the Variable name, you use the IPV4 address. In the case of the Remote environment variables use {{ Ansible\_env.SOME\_VARIABLE }}

**access the Shell Environment Variables**

If you are looking to get access to the existing variables then you can use ‘env’ lookup plugin.Access of the value of Home environment on the management machine would be: local\_home:”{{lookup(‘env’,’HOME’)}}”

write an Ansible ad-hoc command

Ansible [-m ] –a<’arguments’> -u [-become]

• Hosts: it can be any entry within the inventory file. For the specification of all of the hosts in inventory, you may use all or \*. The wild card patterns in this case are also accepted.

• Arguments: you should pass values, which are required by the module. This can change depending on the module, which is used.

• Module name: this is an optional type of parameter. There are many modules, which are available in Ansible. By default, it is command. For one, yup, shell, copy, apt and file.

• Username: it specifies the user account where Ansible may execute commands. The user account, SSH.

• Become: this is one of the optional parameters that are specified when you want to execute operations that require sudo-privilege. By default, become would be false.

• If you put a –c option, then Ansible would do a dry run of the command. It is not going to be actually applied on the nodes.

**disable cowsay within Ansible**

uninstall cowsay or set the Ansible\_NOCOWS environment variable: export Ansible\_NOCOWS= 1(ref Ansible doc).

**copy files recursively on to a target host within Ansible**

In Ansible, the copy module has one of the recursive parameters. On the other hand, you can consider the synchronize module in the event that you want to do something more efficient for a larger number of the files. The synchronize module then wraps resync

**generate crypted passwords when it comes to the user module in Ansible**

The mkpasswd utility which is available on several of the Linux systems is a good alternative:

Mkpasswd – method = sha-512

If this utility is not installed in the system (you may be using OS X) then you may still easily generate the passwords with the use of python. First make sure the Passlib password-hashing library has been installed: Once the library is ready, SHA12 password values may then be generated:

Python –c “from passlib.hash import sha512\_crypt; import getpass; print sha512\_crypt. Using (rounds= 5000). Hash(getpass.getpass())”

Use the integrated Hashing filters for the purpose of generating a hashed version of the password. You may not put plaintext passwords in your playbook or host\_vars; instead, you may use Using Vault in playbooks for encrypting sensitive data. (Ref Ansible doc)

**How do you Access a variable name in a programmatic manner**

{{ hostvars[inventory\_hostname]['Ansible\_' + which\_interface]['ipv4']['address'] }}

The trick when it comes to going through the hostvars is necessary considering it is a dictionary of the namespace of variables. ‘inventory\_hostname’ is a magic variable, which indicates the current host you would be looping over in the host loop

**ask\_sudo\_pass in Ansible**The ask\_sudo\_pass controls the Ansible Playbook in order to prompt the sudo password. At times the default behavior is no:

Ask\_sudo\_pass= true

**ask\_vault\_pass in Ansible**

Using this control, it is possible to determine whether Ansible Playbook would prompt a password for the vault password by default. Usually the default behavior is no:

Ask\_vault\_pass= True

Kubernetes

**Orchestration**

orchestration in the software process means that we can integrate two or more applications. We will be able to automate arrangement, coordination, and management of computer software. The goal of any orchestration process is to streamline and optimize frequent repeatable processes.  
  
6. What is a pod in Kubernetes?

We can think of Kubernetes pod as a group of containers that are run on the same host. So, if we regularly deploy single containers, then our container and the pod will be one and the same.

7. What is a node in Kubernetes?

A node in Kubernetes is a worker machine which is also known as a minion. This node could be a physical machine or a virtual machine. For each node, there is a service to run pods, and it is managed by master components. The node services could include kubelet, kube-proxy, and so on.

Learn more on DevOps in this DevOps Training in Sydney to get ahead in your career!

8. What is a Heapster?

The Heapster lets us do the container cluster monitoring. It lets us do cluster-wide monitoring and event data aggregation. It has native support for Kubernetes.

9. What is a container cluster?

A container cluster lets us place and manage containers in a dynamic setup. It can be considered as a set of nodes or Compute Engine instances. The API server of Kubernetes does not run on cluster nodes, instead the Container Engine hosts the API server.

If you have any doubts or queries related to DevOps, get them clarified from DevOps experts on our DevOps Community!

10. What is a kubelet?

We can think of a kubelet as the lowest level component in a Kubernetes. The kubelet is responsible for making the individual machines run. The sole purpose of a kubelet is that in a given set of containers, it has to ensure that they are all running.  
  
What is Minikube?

The Minikube makes it easy for the local running of Kubernetes. Within a virtual machine, the Minikube runs a single-node Kubernetes cluster.

Interested in becoming a DevOps expert? Click here to learn more in this DevOps Course in Toronto!

12. What is Kubectl?

Kubectl is a Kubernetes command-line tool that is used for deploying and managing applications on Kubernetes. Kubectl is especially useful for inspecting the cluster resources, and for creating, updating, and deleting the components.

13. What is GKE?

GKE is Google Kubernetes Engine which is used for managing and orchestrating systems for Docker containers. GKE also lets us orchestrate container clusters within the Google Public Cloud.

Interested in getting an industry-recognized certification in DevOps? Enroll in Intellipaat’s DevOps Course in Bangalore now!

14. What is kube-proxy?

The kube-proxy runs on each of the nodes. It can do simple tasks such as TCP, UDP, forwarding, and so on. It shows the services in the Kubernetes API on each node.

15. What are the components of a Kubernetes Master?

The components of the Kubernetes Master include the API server, the controller manager, the Scheduler, and the etcd components. The Kubernetes Master components are responsible for running and managing the Kubernetes cluster.

16. What is the use of kube-controller-manager?

It is the Kubernetes Controller Manager. The kube-controller-manager is a daemon that embeds the core control loops which regulates the system state, and it is a non-terminating loop.

Prepare yourself for the DevOps certification with this comprehensive DevOps Training in Hyderabad!

17. What is load balancing on Kubernetes?

The process of load balancing will let us expose services. There are two types of load balancing when it comes to Kubernetes:

Internal load balancing: This is used for balancing the loads automatically and allocating the pods with the required configuration.

External load balancing: This directs the traffic from the external loads to the backend pods.

18. What does a kube-scheduler do?

The kube-scheduler has the job of assigning the nodes to the newly created pods.

19. Where is the Kubernetes cluster data stored?

The primary data store of Kubernetes is etcd, which is responsible for storing all Kubernetes cluster data.

20. How to set a static IP for Kubernetes load balancer?

Kubernetes Master assigns a new IP address.

We can set a static IP for Kubernetes load balancer by changing the DNS records whenever Kubernetes Master assigns a new IP address.

Active Directory

LDAP and Kerberos.

Example: “The Lightweight Directory Access Protocol or LDAP is a protocol that is used to update and query Active Directory. Basically, it is a method that I use to talk to Active Directory as it supports a type of LDAP. The LDAP application protocol can be used for other technologies that provide similar directory services, such as Apache Directory Server. To access objects in Active Directory, LDAP uses two naming paths, which are Distinguished names and Relative Distinguished names.

Kerberos is a key component in Active Directory, as it is the default protocol used for the authentication of all network users. To implement Kerberos by default in a domain or a forest, you need Active Directory Domain Services installed. It boosts the security of the authentication process with cryptography that uses secret keys. Kerberos V5 uses session tickets that can be renewed and encrypted data. It represents an improvement over the challenge/response or NTLM process of authentication, which preceded Kerberos, for u\_\_nlike NTLM, Kerberos does not assume that all servers are genuine.”

**IAM (saml 2.0 and Microsoft Active Directory Federation)**

**Iam groups is a collection of IAM users and Iam roles assigned to users**

Introduction  
  
I am devops engineer. currently working for florida power light at miami.

I work here for a team called cloud enablement team. I provide infrastructure and automate the process of provision for many applications, some of them are big id, talend , streamsets, Alation, Vault and i also work on providing infrastructure for database migration from on prem to cloud. I work on Identiy and Access management in aws. I use terraform and cloudformation for infrastrucutre automation as required by the application teams. I have written terraform modules for different aws services like Lambda, sftps, cloudwatch alarms, IAM roles and Policies.

**Vault**

**I worked on providing infrastructure for vault application and as I deal with Iam. I was involved in setting up the policies in vault. I am familiar with secret engines, aws auth method and policies**

**Ansible use case**

I worked on project called starburst which includes controller and node groups. I have used ansible and used modules, play, playbooks , roles and task and automated the application configuration like boot scripts, installing required packages and monitoring applications for any failures.