

TASK 1 :: Install Terraform

```
arcturus@alpha:~$ wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg
--2024-03-12 06:37:42-- https://apt.releases.hashicorp.com/gpg
Resolving apt.releases.hashicorp.com (apt.releases.hashicorp.com)... [sudo] password for arcturus: 108.158.221.60, 108.158.221.108, 108.158.221.87, ...
Connecting to apt.releases.hashicorp.com (apt.releases.hashicorp.com)[108.158.221.60]:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3980 (3.9K) [binary/octet-stream]
Saving to: 'STDOUT'

-
=====] 3.89K --.-KB/s in 0s 100%[=====]
2024-03-12 06:37:42 (60.1 MB/s) - written to stdout [3980/3980]

^[[ASorry, try again.
[sudo] password for arcturus:
Sorry, try again.
[sudo] password for arcturus:
File '/usr/share/keyrings/hashicorp-archive-keyring.gpg' exists. Overwrite? (y/N) y
arcturus@alpha:~$ echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb_release -cs) main" | sudo
tee /etc/apt/sources.list.d/hashicorp.list
deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com jammy main
arcturus@alpha:~$ sudo apt update && sudo apt install terraform
Hit:1 https://apt.releases.hashicorp.com jammy InRelease
Hit:2 https://download.docker.com/linux/ubuntu jammy InRelease
Ign:3 https://apt.datadoghq.com stable InRelease
Hit:4 https://deb.nodesource.com/node_19.x jammy InRelease
Hit:5 https://apt.datadoghq.com stable Release
Hit:6 https://dl.google.com/linux/chrome/deb stable InRelease
Hit:7 http://in.archive.ubuntu.com/ubuntu jammy InRelease
Hit:8 https://baltoctdn.com/helm/stable/debian all InRelease
Hit:9 http://in.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:10 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:11 http://in.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:12 https://ppa.launchpadcontent.net/libratbag-piper/piper-libratbag-git/ubuntu jammy InRelease
Hit:13 https://ngrok-agent.s3.amazonaws.com buster InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done

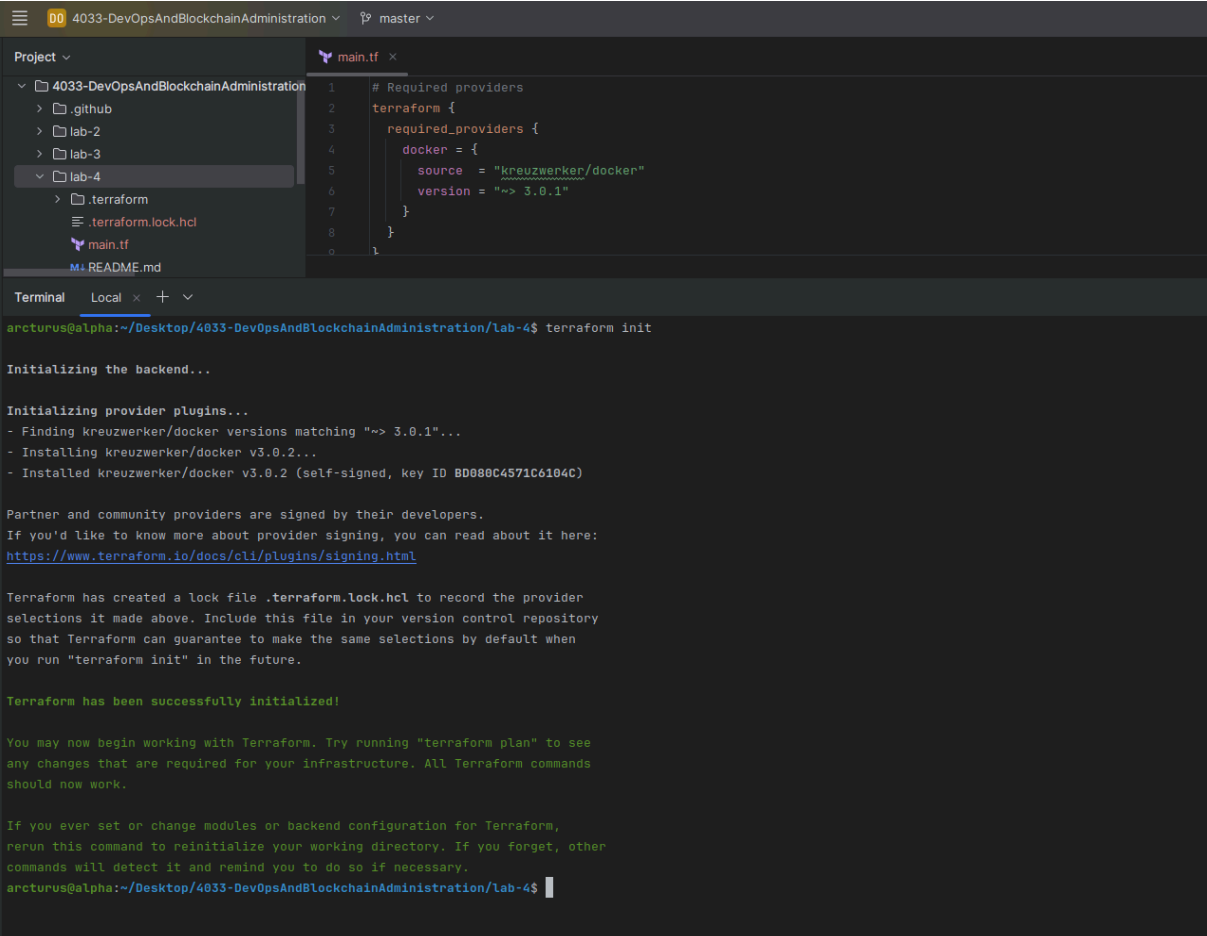
38 packages can be upgraded. Run 'apt list --upgradable' to see them.
W: https://download.docker.com/linux/ubuntu/dists/jammy/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION sec
tion in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libqt5help5 libqt5sql5 libqt5sql5-sqlite libqt5xml5
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  terraform
0 upgraded, 1 newly installed, 0 to remove and 38 not upgraded.
Need to get 26.9 MB of archives.
After this operation, 84.5 MB of additional disk space will be used.
Get:1 https://apt.releases.hashicorp.com jammy/main amd64 terraform amd64 1.7.4-1 [26.9 MB]
Fetched 26.9 MB in 1s (23.5 MB/s)
Selecting previously unselected package terraform.
(Reading database ... 323753 files and directories currently installed.)
Preparing to unpack .../terraform-1.7.4-1_amd64.deb ...
Unpacking terraform (1.7.4-1) ...
Setting up terraform (1.7.4-1) ...
arcturus@alpha:~$ terraform --version
Terraform v1.7.4
on linux_amd64
arcturus@alpha:~$
```

TASK 2 :: Create a tf configuration for running docker resource

```
main.tf x
1  # Required providers
2  terraform {
3      required_providers {
4          docker = {
5              source = "kreuzwerker/docker"
6              version = "~> 3.0.1"
7          }
8      }
9  }
10
11 # Docker provider
12 provider "docker" {}
13
14 # MongoDB container
15 resource "docker_container" "mongodb" {
16     name     = "mongodb"
17     image    = "mongo:latest"
18     restart  = "always"
19 }
20
21 # Node.js application container
22 resource "docker_container" "nodejs" {
23     name     = "nodejs"
24     image    = "node:latest"
25     restart  = "always"
26
27     depends_on = [docker_container.mongodb]
28 }
29
30 # Nginx server container
31 resource "docker_container" "nginx" {
32     name     = "nginx"
33     image    = "nginx:latest"
34     restart  = "always"
35     ports {
36         internal = "80"
37         external = "8080"
38     }
39
40     depends_on = [docker_container.nodejs]
41 }
42
```

TASK 3 :: Apply the initial resources

Terraform init



The screenshot shows a code editor with a project named "4033-DevOpsAndBlockchainAdministration" on the "master" branch. The file explorer on the left shows a directory structure with "lab-4" selected, containing ".terraform", ".terraform.lock.hcl", "main.tf", and "README.md". The editor displays the content of "main.tf", which is a Terraform configuration for required providers. The terminal window at the bottom shows the execution of the "terraform init" command, which initializes the backend and installs the required providers.

```
1 # Required providers
2 terraform {
3   required_providers {
4     docker = {
5       source = "kreuzwerker/docker"
6       version = "~> 3.0.1"
7     }
8   }
9 }
```

Terminal output:

```
arecturus@alpha:~/Desktop/4033-DevOpsAndBlockchainAdministration/lab-4$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding kreuzwerker/docker versions matching "~> 3.0.1"...
- Installing kreuzwerker/docker v3.0.2...
- Installed kreuzwerker/docker v3.0.2 (self-signed, key ID BD080C4571C6104C)

Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html

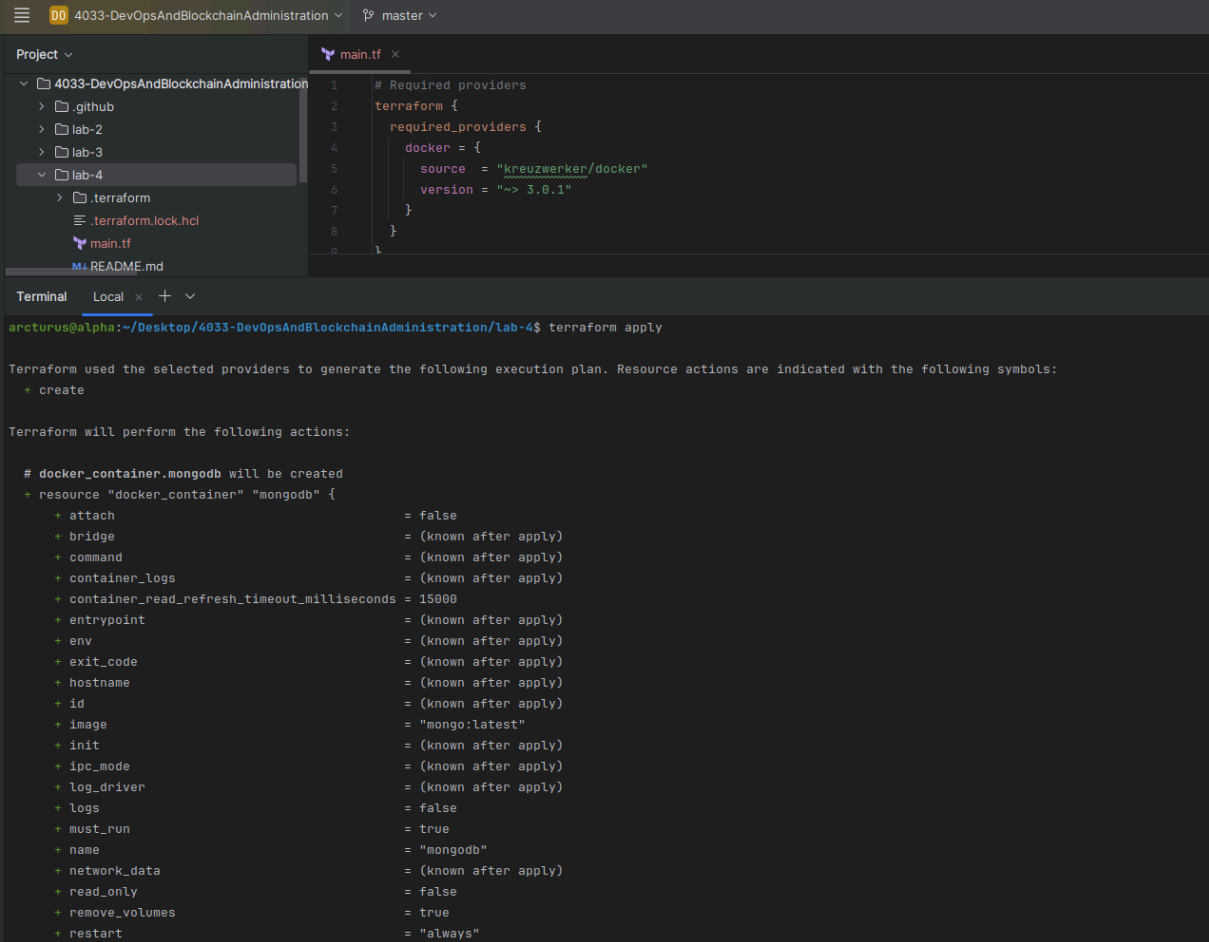
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
arecturus@alpha:~/Desktop/4033-DevOpsAndBlockchainAdministration/lab-4$
```

Terraform Apply



The screenshot shows a code editor with a project named "4033-DevOpsAndBlockchainAdministration". The file explorer on the left shows a directory structure with "lab-4" selected, containing ".terraform", ".terraform.lock.hcl", "main.tf", and "README.md". The "main.tf" file is open, showing a Terraform configuration for a Docker container named "mongodb".

```
1 # Required providers
2 terraform {
3   required_providers {
4     docker = {
5       source  = "kreuzwerker/docker"
6       version = "~> 3.0.1"
7     }
8   }
9 }
```

The terminal window shows the command `terraform apply` being executed. The output indicates that Terraform will create a resource named "docker_container.mongodb".

```
arcturus@Alpha:~/Desktop/4033-DevOpsAndBlockchainAdministration/lab-4$ terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# docker_container.mongodb will be created
+ resource "docker_container" "mongodb" {
  + attach                = false
  + bridge                = (known after apply)
  + command               = (known after apply)
  + container_logs        = (known after apply)
  + container_read_refresh_timeout_milliseconds = 15000
  + endpoint              = (known after apply)
  + env                  = (known after apply)
  + exit_code             = (known after apply)
  + hostname              = (known after apply)
  + id                   = (known after apply)
  + image                 = "mongo:latest"
  + init                  = (known after apply)
  + ipc_mode              = (known after apply)
  + log_driver            = (known after apply)
  + logs                  = false
  + must_run              = true
  + name                  = "mongodb"
  + network_data          = (known after apply)
  + read_only             = false
  + remove_volumes        = true
  + restart               = "always"
```

```
# docker_container.nginx will be created
+ resource "docker_container" "nginx" {
  + attach                = false
  + bridge                = (known after apply)
  + command               = (known after apply)
  + container_logs        = (known after apply)
  + container_read_refresh_timeout_milliseconds = 15000
  + entrypoint            = (known after apply)
  + env                   = (known after apply)
  + exit_code             = (known after apply)
  + hostname              = (known after apply)
  + id                    = (known after apply)
  + image                  = "nginx:latest"
  + init                  = (known after apply)
  + ipc_mode              = (known after apply)
  + log_driver            = (known after apply)
  + logs                  = false
  + must_run              = true
  + name                  = "nginx"
  + network_data          = (known after apply)
  + read_only             = false
  + remove_volumes       = true
  + restart               = "always"
  + rm                    = false
  + runtime               = (known after apply)
  + security_opts         = (known after apply)
  + shm_size              = (known after apply)
  + start                 = true
  + stdin_open            = false
  + stop_signal           = (known after apply)
  + stop_timeout          = (known after apply)
  + tty                   = false
  + wait                  = false
  + wait_timeout          = 60

  + ports {
    + external = 8080
    + internal = 80
    + ip       = "0.0.0.0"
    + protocol = "tcp"
  }
}
```

```
  + protocol = "tcp"
}
}

# docker_container.nodejs will be created
+ resource "docker_container" "nodejs" {
  + attach                = false
  + bridge                = (known after apply)
  + command               = (known after apply)
  + container_logs        = (known after apply)
  + container_read_refresh_timeout_milliseconds = 15000
  + entrypoint            = (known after apply)
  + env                   = (known after apply)
  + exit_code             = (known after apply)
  + hostname              = (known after apply)
  + id                    = (known after apply)
  + image                  = "node:latest"
  + init                  = (known after apply)
```

```

+ name = "nodejs"
+ network_data = (known after apply)
+ read_only = false
+ remove_volumes = true
+ restart = "always"
+ rm = false
+ runtime = (known after apply)
+ security_opts = (known after apply)
+ shm_size = (known after apply)
+ start = true
+ stdin_open = false
+ stop_signal = (known after apply)
+ stop_timeout = (known after apply)
+ tty = false
+ wait = false
+ wait_timeout = 60
}

Plan: 3 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

docker_container.mongodb: Creating...
docker_container.mongodb: Creation complete after 0s [id=f2be3b1819ca1e9869810e83adeef897bf6ec8dc6b4e4264ab9202e4f57a5e01]
docker_container.nodejs: Creating...
docker_container.nodejs: Still creating... [10s elapsed]
docker_container.nodejs: Still creating... [20s elapsed]
docker_container.nodejs: Still creating... [30s elapsed]
docker_container.nodejs: Still creating... [40s elapsed]
docker_container.nodejs: Creation complete after 41s [id=0ebbd24507e25efad2b057fbd4659c0f22a4c5c76440820d9ad43ca442409392]
docker_container.nginx: Creating...
docker_container.nginx: Still creating... [10s elapsed]
docker_container.nginx: Still creating... [20s elapsed]
docker_container.nginx: Creation complete after 22s [id=e7d2e9ef38132fb6d576088c3548697dbaa022ba9887376e8de827f599bb99ae]

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.
arcturus@alpha:~/Desktop/4033-DevOpsAndBlockchainAdministration/lab-4$

```

TASK 4 :: Change infrastructure (Changed the external port of the Nginx server from 8080 to 8081)

```

main.tf x
19  }
20
21  # Node.js application container
22  resource "docker_container" "nodejs" {
23    name = "nodejs"
24    image = "node:latest"
25    restart = "always"
26
27    depends_on = [docker_container.mongodb]
28  }
29
30  # Nginx server container
31  resource "docker_container" "nginx" {
32    name = "nginx"
33    image = "nginx:latest"
34    restart = "always"
35    ports {
36      internal = "80"
37      external = "8081"
38    }
39
40    depends_on = [docker_container.nodejs]
41  }
42

```

TASK 5 :: Create a plan and use the plan to make changes to the resource

```
arcturus@alpha:~/Desktop/4033-DevOpsAndBlockchainAdministration/lab-4$ terraform plan -out=tfplan
docker_container.mongodb: Refreshing state... [id=f2be3b1819ca1e9869810e83adeef897bf6ec8dc6b4e4264ab9202e4f57a5e01]
docker_container.nodejs: Refreshing state... [id=0ebbd24507e25efad2b057fbd4659c0f22a4c5c76440820d9ad43ca442409392]
docker_container.nginx: Refreshing state... [id=e7d2e9ef38132fb6d576088c3548697dbaa022ba9887376e8de827f599bb99ae]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

# docker_container.mongodb must be replaced
/+ resource "docker_container" "mongodb" {
  + bridge              = (known after apply)
  + command             = [
    - "mongod",
  ] -> (known after apply)
  + container_logs      = (known after apply)
  - cpu_shares          = 0 -> null
  - dns                 = [] -> null
  - dns_opts            = [] -> null
  - dns_search          = [] -> null
  + entrypoint          = [
    - "docker-entrypoint.sh",
  ] -> (known after apply)
  + env                 = [] -> (known after apply)
  + exit_code           = (known after apply)
  - group_add           = [] -> null
  + hostname            = "f2be3b1819ca" -> (known after apply)
  + id                  = "f2be3b1819ca1e9869810e83adeef897bf6ec8dc6b4e4264ab9202e4f57a5e01" -> (known after apply)
  + image               = "sha256:b8df2163f9aa384163ea74e076f8cf502f8d291189dbdecc79036e546e1a989c" -> "mongo:latest" # forces replacement
  + init                = false -> (known after apply)
  + ipc_mode            = "private" -> (known after apply)
  + log_driver          = "json-file" -> (known after apply)
  - log_opts            = {} -> null
  - max_retry_count     = 0 -> null
  - memory              = 0 -> null
  - memory_swap         = 0 -> null
  + name                = "mongodb"
  + network_data        = [
    - {
      - log_opts              = {} -> null
      - max_retry_count      = 0 -> null
      - memory               = 0 -> null
      - memory_swap         = 0 -> null
      + name                 = "mongodb"
      + network_data         = [
        - {
          - gateway          = "172.17.0.1"
          - global_ipv6_address = ""
          - global_ipv6_prefix_length = 0
          - ip_address       = "172.17.0.4"
          - ip_prefix_length = 16
          - ipv6_gateway     = ""
          - mac_address      = "02:42:ac:11:00:04"
          - network_name     = "bridge"
        },
      ] -> (known after apply)
      - network_mode        = "default" -> null
      - privileged          = false -> null
      - publish_all_ports   = false -> null
      + runtime             = "runc" -> (known after apply)
      + security_opts        = [] -> (known after apply)
      + shm_size            = 64 -> (known after apply)
      + stop_signal          = (known after apply)
      + stop_timeout        = 0 -> (known after apply)
      - storage_opts        = {} -> null
      - sysctls             = {} -> null
      - tmpfs               = {} -> null
    }
  ]
}

# docker_container.nginx must be replaced
/+ resource "docker_container" "nginx" {
  + bridge              = (known after apply)
  + command             = [
    - "nginx",
    - "-g",
    - "daemon off;",
  ] -> (known after apply)
  + container_logs      = (known after apply)
```

```
- log_opts              = {} -> null
- max_retry_count      = 0 -> null
- memory               = 0 -> null
- memory_swap         = 0 -> null
+ name                 = "mongodb"
+ network_data         = [
  - {
    - gateway          = "172.17.0.1"
    - global_ipv6_address = ""
    - global_ipv6_prefix_length = 0
    - ip_address       = "172.17.0.4"
    - ip_prefix_length = 16
    - ipv6_gateway     = ""
    - mac_address      = "02:42:ac:11:00:04"
    - network_name     = "bridge"
  },
] -> (known after apply)
- network_mode        = "default" -> null
- privileged          = false -> null
- publish_all_ports   = false -> null
+ runtime             = "runc" -> (known after apply)
+ security_opts        = [] -> (known after apply)
+ shm_size            = 64 -> (known after apply)
+ stop_signal          = (known after apply)
+ stop_timeout        = 0 -> (known after apply)
- storage_opts        = {} -> null
- sysctls             = {} -> null
- tmpfs               = {} -> null
# (13 unchanged attributes hidden)
}

# docker_container.nginx must be replaced
/+ resource "docker_container" "nginx" {
  + bridge              = (known after apply)
  + command             = [
    - "nginx",
    - "-g",
    - "daemon off;",
  ] -> (known after apply)
  + container_logs      = (known after apply)
```

```

- "-g",
- "daemon off;",
] -> (known after apply)
+ container_logs = (known after apply)
- cpu_shares = 0 -> null
- dns = [] -> null
- dns_opts = [] -> null
- dns_search = [] -> null
+ entrypoint = [
- "/docker-entrypoint.sh",
] -> (known after apply)
+ env = [] -> (known after apply)
+ exit_code = (known after apply)
- group_add = [] -> null
+ hostname = "e7d2e9ef3813" -> (known after apply)
+ id = "e7d2e9ef38132fb6d576088c3548697dbaa022ba9887376e8de827f599bb99ae" -> (known after apply)
+ image = "sha256:e4720093a3c1381245b53a5a51b417963b3c4472d3f47fc301930a4f3b17666a" -> "nginx:latest" # forces replacement
+ init = false -> (known after apply)
+ ipc_mode = "private" -> (known after apply)
+ log_driver = "json-file" -> (known after apply)
- log_opts = {} -> null
- max_retry_count = 0 -> null
- memory = 0 -> null
- memory_swap = 0 -> null
+ name = "nginx"
+ network_data = [
- {
- gateway = "172.17.0.1"
- global_ipv6_address = ""
- global_ipv6_prefix_length = 0
- ip_address = "172.17.0.5"
- ip_prefix_length = 16
- ipv6_gateway = ""
- mac_address = "02:42:ac:11:00:05"
- network_name = "bridge"
},
] -> (known after apply)
- network_mode = "default" -> null
- privileged = false -> null
- publish_all_ports = false -> null

```

```

- publish_all_ports = false -> null
+ runtime = "runc" -> (known after apply)
+ security_opts = [] -> (known after apply)
+ shm_size = 64 -> (known after apply)
+ stop_signal = "SIGQUIT" -> (known after apply)
+ stop_timeout = 0 -> (known after apply)
- storage_opts = {} -> null
- sysctls = {} -> null
- tmpfs = {} -> null
# (13 unchanged attributes hidden)

+ ports {
+ external = 8080 -> 8081 # forces replacement
# (3 unchanged attributes hidden)
}
}

# docker_container.nodejs must be replaced
/* resource "docker_container" "nodejs" {
+ bridge = (known after apply)
+ command = [
- "node",
] -> (known after apply)
+ container_logs = (known after apply)
- cpu_shares = 0 -> null
- dns = [] -> null
- dns_opts = [] -> null
- dns_search = [] -> null
+ entrypoint = [
- "docker-entrypoint.sh",
] -> (known after apply)
+ env = [] -> (known after apply)
+ exit_code = (known after apply)
- group_add = [] -> null
+ hostname = "0ebbd24507e2" -> (known after apply)
+ id = "0ebbd24507e25efad2b057fbd4659c0f22a4c5c76440820d9ad43ca442409392" -> (known after apply)
+ image = "sha256:8bb9d1bc636e182ccb83c6961219c80e4a56e0bd8b205a4ab126680030f7745" -> "node:latest" # forces replacement
+ init = false -> (known after apply)
+ ipc_mode = "private" -> (known after apply)
+ log_driver = "json-file" -> (known after apply)

```



```

- memory = 0 -> null
- memory_swap = 0 -> null
- name = "nodejs"
- network_data = [
  - {
    - gateway = ""
    - global_ipv6_address = ""
    - global_ipv6_prefix_length = 0
    - ip_address = ""
    - ip_prefix_length = 0
    - ipv6_gateway = ""
    - mac_address = ""
    - network_name = "bridge"
  },
] -> (known after apply)
- network_mode = "default" -> null
- privileged = false -> null
- publish_all_ports = false -> null
- runtime = "runc" -> (known after apply)
- security_opts = [] -> (known after apply)
- shm_size = 64 -> (known after apply)
+ stop_signal = (known after apply)
- stop_timeout = 0 -> (known after apply)
- storage_opts = {} -> null
- sysctls = {} -> null
- tmpfs = {} -> null
# (13 unchanged attributes hidden)
}

```

Plan: 3 to add, 0 to change, 3 to destroy.

Saved the plan to: tfplan

To perform exactly these actions, run the following command to apply:

```

terraform apply "tfplan"
arcturus@alpha:~/Desktop/4033-DevOpsAndBlockchainAdministration/lab-4$

```

```

- sysctls = {} -> null
- tmpfs = {} -> null
# (13 unchanged attributes hidden)
}

```

Plan: 3 to add, 0 to change, 3 to destroy.

Saved the plan to: tfplan

To perform exactly these actions, run the following command to apply:

```

terraform apply "tfplan"
arcturus@alpha:~/Desktop/4033-DevOpsAndBlockchainAdministration/lab-4$ terraform apply tfplan
docker_container.nginx: Destroying... [id=e7d2e9ef38132fb6d576088c3548697dbaa022ba9887376e8de827f599bb99ae]
docker_container.nginx: Destruction complete after 1s
docker_container.nodejs: Destroying... [id=0ebbd24507e25efad2b057fbd4659c0f22a4c5c76440820d9ad43ca42409392]
docker_container.nodejs: Destruction complete after 0s
docker_container.mongodb: Destroying... [id=f2be3b1819ca1e9869810e83adeef897bf6ec8dc6b4e4264ab9202e4f57a5e01]
docker_container.mongodb: Destruction complete after 0s
docker_container.mongodb: Creating...
docker_container.mongodb: Creation complete after 1s [id=588588f06d12a08602abfe1b43440d8663f34ca380e9ac61be64c048be69f210]
docker_container.nodejs: Creating...
docker_container.nodejs: Creation complete after 0s [id=718b424af368dce59eef76d2c6ed2362c673322d1d90d2874fb24e07e92d0c3e]
docker_container.nginx: Creating...
docker_container.nginx: Creation complete after 1s [id=ee764bf117ba49e0448f3b30a68bfc96cd479f88ff837e39c461137279f77e7c]

```

Apply complete! Resources: 3 added, 0 changed, 3 destroyed.

```

arcturus@alpha:~/Desktop/4033-DevOpsAndBlockchainAdministration/lab-4$

```

TASK 6 :: Make destructive changes (Removed MongoDB Container)

```
main.tf x
1  # Required providers
2  terraform {
3      required_providers {
4          docker = {
5              source = "kreuzwerker/docker"
6              version = "~> 3.0.1"
7          }
8      }
9  }
10
11  # Docker provider
12  provider "docker" {}
13
14  # MongoDB container
15  #resource "docker_container" "mongodb" {
16  #   name      = "mongodb"
17  #   image     = "mongo:latest"
18  #   restart   = "always"
19  #}
20
21  # Node.js application container
22  resource "docker_container" "nodejs" {
23      name      = "nodejs"
24      image     = "node:latest"
25      restart   = "always"
26
27      # depends_on = [docker_container.mongodb]
28  }
29
30  # Nginx server container
31  resource "docker_container" "nginx" {
```



```

# docker_container.nginx must be replaced
-/+ resource "docker_container" "nginx" {
  + bridge                               = (known after apply)
  ~ command                             = [
    - "nginx",
    - "-g",
    - "daemon off;",
  ] -> (known after apply)
  + container_logs                       = (known after apply)
  - cpu_shares                           = 0 -> null
  - dns                                  = [] -> null
  - dns_opts                             = [] -> null
  - dns_search                           = [] -> null
  ~ endpoint                             = [
    - "/docker-entrypoint.sh",
  ] -> (known after apply)
  ~ env                                  = [] -> (known after apply)
  + exit_code                            = (known after apply)
  - group_add                             = [] -> null
  ~ hostname                             = "ee764bf117ba" -> (known after apply)
  ~ id                                    = "ee764bf117ba49e0448f3b30a68bfc96cd479f88ff837e39c461137279f77e7c" -> (known after apply)
  ~ image                                 = "sha256:e4720093a3c1381245b53a5a51b417963b3c4472d3f47fc301930a4f3b17666a" -> "nginx:latest" # forces replacement
  ~ init                                 = false -> (known after apply)
  ~ ipc_mode                             = "private" -> (known after apply)
  ~ log_driver                           = "json-file" -> (known after apply)
  - log_opts                             = {} -> null
  - max_retry_count                       = 0 -> null
  - memory                               = 0 -> null
  - memory_swap                          = 0 -> null
  ~ name                                  = "nginx"
  ~ network_data                         = [
    - {
      - gateway                           = "172.17.0.1"
      - global_ipv6_address               = ""
      - global_ipv6_prefix_length         = 0
      - ip_address                        = "172.17.0.6"
      - ip_prefix_length                  = 16
      - ipv6_gateway                      = ""
      - mac_address                       = "02:42:ac:11:00:06"
    }
  ]
}

```

```

- network_mode                           = "default" -> null
- privileged                             = false -> null
- publish_all_ports                      = false -> null
~ runtime                                = "runc" -> (known after apply)
~ security_opts                          = [] -> (known after apply)
~ shm_size                               = 64 -> (known after apply)
~ stop_signal                            = "SIGQUIT" -> (known after apply)
~ stop_timeout                           = 0 -> (known after apply)
- storage_opts                           = {} -> null
- sysctls                                = {} -> null
- tmpfs                                  = {} -> null
# (13 unchanged attributes hidden)

# (1 unchanged block hidden)
}

# docker_container.nodejs must be replaced
-/+ resource "docker_container" "nodejs" {
  + bridge                               = (known after apply)
  ~ command                             = [
    - "node",
  ] -> (known after apply)
  + container_logs                       = (known after apply)
  - cpu_shares                           = 0 -> null
  - dns                                  = [] -> null
  - dns_opts                             = [] -> null
  - dns_search                           = [] -> null
  ~ endpoint                             = [
    - "docker-entrypoint.sh",
  ] -> (known after apply)
  ~ env                                  = [] -> (known after apply)
  + exit_code                            = (known after apply)
  - group_add                             = [] -> null
  ~ hostname                             = "718b424af368" -> (known after apply)
  ~ id                                    = "718b424af368dce59eef76d2c0ed2362c673322d1d90d2874fb24e07e92d0c3e" -> (known after apply)
  ~ image                                 = "sha256:8bbb9d1bc633e182ccb83c6961219c80e4a56e0bd8b205a4ab126680030f7745" -> "node:latest" # forces replacement
  ~ init                                 = false -> (known after apply)
  ~ ipc_mode                             = "private" -> (known after apply)
  ~ log_driver                           = "json-file" -> (known after apply)
}

```

```

- mac_address          = ""
- network_name         = "bridge"
},
] -> (known after apply)
- network_mode         = "default" -> null
- privileged           = false -> null
- publish_all_ports    = false -> null
- runtime              = "runc" -> (known after apply)
- security_opts        = [] -> (known after apply)
- shm_size             = 64 -> (known after apply)
+ stop_signal          = (known after apply)
- stop_timeout         = 0 -> (known after apply)
- storage_opts         = {} -> null
- sysctls              = {} -> null
- tmpfs               = {} -> null
# (13 unchanged attributes hidden)
}

Plan: 2 to add, 0 to change, 3 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

docker_container.nginx: Destroying... [id=ee764bf117ba49e0448f3b30a60bfc96cd479f88ff837e39c461137279f77e7c]
docker_container.nginx: Destruction complete after 0s
docker_container.nodejs: Destroying... [id=718b424af368dce59eef76d2c6ed2362c673322d1d90d2874fb24e07e92d0c3e]
docker_container.nodejs: Destruction complete after 0s
docker_container.mongodb: Destroying... [id=588588f06d12a08602abfe1b43440d8663f34ca380e9ac61be64c048be69f210]
docker_container.mongodb: Destruction complete after 1s
docker_container.nodejs: Creating...
docker_container.nodejs: Creation complete after 0s [id=6d9bbe1f4f5c4d205e9e0e30cab5bdf566870a2bc7452ce891ea04fad0ee20e0]
docker_container.nginx: Creating...
docker_container.nginx: Creation complete after 1s [id=ff3411968707f1999df5da128d657e2669525be42d9d2216aa97b0814ea65289]

Apply complete! Resources: 2 added, 0 changed, 3 destroyed.
arcturus@alpha:~/Desktop/4033-DevOpsAndBlockchainAdministration/lab-4$

```

TASK 7 :: Destroy the complete resource

```

arcturus@alpha:~/Desktop/4033-DevOpsAndBlockchainAdministration/lab-4$ terraform destroy
docker_container.nodejs: Refreshing state... [id=6d9bbe1f4f5c4d205e9e0e30cab5bdf566870a2bc7452ce891ea04fad0ee20e0]
docker_container.nginx: Refreshing state... [id=ff3411968707f1999df5da128d657e2669525be42d9d2216aa97b0814ea65289]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
- destroy

Terraform will perform the following actions:

# docker_container.nginx will be destroyed
- resource "docker_container" "nginx" {
  - attach          = false -> null
  - command         = [
    - "nginx",
    - "-g",
    - "daemon off;",
  ] -> null
  - container_read_refresh_timeout_milliseconds = 15000 -> null
  - cpu_shares      = 0 -> null
  - dns             = [] -> null
  - dns_opts       = [] -> null
  - dns_search     = [] -> null
  - entrypoint      = [
    - "/docker-entrypoint.sh",
  ] -> null
  - env            = [] -> null
  - group_add      = [] -> null
  - hostname       = "ff3411968707" -> null
  - id             = "ff3411968707f1999df5da128d657e2669525be42d9d2216aa97b0814ea65289" -> null
  - image          = "sha256:e4720893a3c1381245b53a5a51b417963b3c4472d3f47fc301930a4f3b17666a" -> null
  - init           = false -> null
  - ipc_mode       = "private" -> null
  - log_driver      = "json-file" -> null
  - log_opts       = {} -> null
  - logs           = false -> null
  - max_retry_count = 0 -> null
  - memory         = 0 -> null
  - memory_swap    = 0 -> null
}

```

```

- memory = 0 -> null
- memory_swap = 0 -> null
- must_run = true -> null
- name = "nginx" -> null
- network_data = [
  - {
    - gateway = "172.17.0.1"
    - global_ipv6_address = ""
    - global_ipv6_prefix_length = 0
    - ip_address = "172.17.0.5"
    - ip_prefix_length = 16
    - ipv6_gateway = ""
    - mac_address = "02:42:ac:11:00:05"
    - network_name = "bridge"
  },
] -> null
- network_mode = "default" -> null
- privileged = false -> null
- publish_all_ports = false -> null
- read_only = false -> null
- remove_volumes = true -> null
- restart = "always" -> null
- rm = false -> null
- runtime = "runc" -> null
- security_opts = [] -> null
- shm_size = 64 -> null
- start = true -> null
- stdin_open = false -> null
- stop_signal = "SIGQUIT" -> null
- stop_timeout = 0 -> null
- storage_opts = {} -> null
- sysctls = {} -> null
- tmpfs = {} -> null
- tty = false -> null
- wait = false -> null
- wait_timeout = 60 -> null

- ports {

```

```

# docker_container.nodejs will be destroyed
- resource "docker_container" "nodejs" {
  - attach = false -> null
  - command = [
    - "node",
  ] -> null
  - container_read_refresh_timeout_milliseconds = 15000 -> null
  - cpu_shares = 0 -> null
  - dns = [] -> null
  - dns_opts = [] -> null
  - dns_search = [] -> null
  - entrypoint = [
    - "docker-entrypoint.sh",
  ] -> null
  - env = [] -> null
  - group_add = [] -> null
  - hostname = "6d9bbe1f4f5c" -> null
  - id = "6d9bbe1f4f5c4d205e9e0e30cab5bdf566870a2bc7452ce891ea04fad0ee20e0" -> null
  - image = "sha256:8bbb9d1bc633e182ccb83c6961219c80e4a56e0bd8b205a4ab126680030f7745" -> null
  - init = false -> null
  - ipc_mode = "private" -> null
  - log_driver = "json-file" -> null
  - log_opts = {} -> null
  - logs = false -> null
  - max_retry_count = 0 -> null
  - memory = 0 -> null
  - memory_swap = 0 -> null
  - must_run = true -> null
  - name = "nodejs" -> null
  - network_data = [
    - {
      - gateway = ""
      - global_ipv6_address = ""
      - global_ipv6_prefix_length = 0
      - ip_address = ""
      - ip_prefix_length = 0
      - ipv6_gateway = ""
      - mac_address = ""
    }
  ]
}

```

```

    },
    ] -> null
- network_mode                = "default" -> null
- privileged                  = false -> null
- publish_all_ports           = false -> null
- read_only                   = false -> null
- remove_volumes              = true -> null
- restart                     = "always" -> null
- rm                           = false -> null
- runtime                     = "runc" -> null
- security_opts                = [] -> null
- shm_size                     = 64 -> null
- start                        = true -> null
- stdin_open                   = false -> null
- stop_timeout                 = 0 -> null
- storage_opts                 = {} -> null
- sysctls                     = {} -> null
- tmpfs                        = {} -> null
- tty                          = false -> null
- wait                         = false -> null
- wait_timeout                 = 60 -> null
}

Plan: 0 to add, 0 to change, 2 to destroy.

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

docker_container.nginx: Destroying... [id=ff3411968707f1999df5da128d657e2669525be42d9d2216aa97b8814ea65289]
docker_container.nginx: Destruction complete after 0s
docker_container.nodejs: Destroying... [id=6d9bbe1f4f5c4d205e9e0e30cab5bdf566870a2bc7452ce891ea04fad0ee28e0]
docker_container.nodejs: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.
arcturus@alpha:~/Desktop/4033-DevOpsAndBlockchainAdministration/lab-4$

```

TASK 8 :: Create resource with dependencies (implicit and explicit)

The reference to the `docker_container.mongodb` resource in the `docker_container.nodejs` resource creates an implicit dependency.

```

# MongoDB container
resource "docker_container" "mongodb" {
    name     = "mongodb"
    image    = "mongo:latest"
    restart  = "always"
}

# Node.js application container
resource "docker_container" "nodejs" {
    name     = "nodejs"
    image    = "node:latest"
    restart  = "always"
    ⚡
    depends_on = [docker_container.mongodb]
}

```

The `depends_on` field in the `docker_container.nodejs` and `docker_container.nginx` resources creates explicit dependencies.

```
main.tf x
4     docker = {
5         source = "kreuzwerker/docker"
6         version = "~> 3.0.1"
7     }
8 }
9
10
11 # Docker provider
12 provider "docker" {}
13
14 # MongoDB container
15 resource "docker_container" "mongodb" {
16     name     = "mongodb"
17     image    = "mongo:latest"
18     restart  = "always"
19 }
20
21 # Node.js application container
22 resource "docker_container" "nodejs" {
23     name     = "nodejs"
24     image    = "node:latest"
25     restart  = "always"
26
27     depends_on = [docker_container.mongodb]
28 }
29
30 # Nginx server container
31 resource "docker_container" "nginx" {
32     name     = "nginx"
33     image    = "nginx:latest"
34     restart  = "always"
35     ports {
36         internal = "80"
37         external = "8081"
38     }
39
40     depends_on = [docker_container.nodejs]
41 }
42
43
```