

CORE WORKFLOW

```
terraform init # provider
s, modules, backend
terraform fmt -recursive -check
terraform validate
terraform plan -out=tfplan
terraform apply tfplan
terraform destroy
```

- init:** `-upgrade`, `-reconfigure`, `-migrate-state`, `-backend-config=f.hcl`, `-lockfile=readonly`.
- plan/apply:** `-var a=b`, `-var-file=prod.tfvars`, `-replace=ADDR`, `-target=ADDR` (exceptional), `-refresh-only`, `-minimal-refresh` (1.17), `-destroy`, `-detailed-exitcode` (0 none, 1 error, 2 changes), `-input=false`, `-lock-timeout=5m`, `-parallelism=10`, `-auto-approve`.
- Saved plan files contain secrets: treat them like state.
- Commit `.terraform.lock.hcl`; never commit `.terraform/`, `*.tfstate`.

CLI

```
terraform state list | show ADDR
| mv A B
terraform state rm ADDR | pull |
push
terraform import ADDR ID #
legacy
terraform query #
i.14+
terraform workspace new|select|li
st dev
terraform providers local -platfor
m=linux_amd64
terraform providers schema -json
terraform test -verbose
terraform console #
try exprs
terraform graph | dot -Tsvg > g.s
vg
terraform output -json | -raw nam
e
terraform show -json tfplan
terraform force-unlock LOCK_ID
terraform login ; terraform versi
on -json
```

Env vars: `TF_VAR_name`, `TF_LOG` (TRACE, DEBUG, INFO, WARN, ERROR, JSON), `TF_LOG_PATH`, `TF_LOG_CORE/TF_LOG_PROVIDER`, `TF_INPUT=0`, `TF_IN_AUTOMATION`, `TF_CLI_ARGS(p_name)`, `TF_DATA_DIR`, `TF_WORKSPACE`, `TF_PLUGIN_CACHE_DIR`, `TF_CLI_CONFIG_FILE`, `TF_TOKEN_app_terraform_io`.

HCL SKELETON

```
terraform {
  required_version = ">= 1.17"
  required_providers {
    aws = { source = "hashicorp/a
ws", version = ">= 6.0" }
  }
  backend "s3" { ... }
}
provider "aws" { region = var.reg
ion }
resource "aws_s3_bucket" "covers" {
  bucket = "${local.prefix}-cover
s"
}
data "aws_caller_identity" "me" {}
variable "region" {
  type = string
  default = "eu-west-1"
}
output "bucket" { value = aws_s3_
bucket.covers.id }
locals { prefix = "bookshelf-${te
rraform.workspace}" }
module "net" {
  source = "terraform-aws-module
s/vpc/aws"
  version = ">= 6.0"
  cidr = "10.0.0.0/16"
}
```

Files: `main.tf`, `variables.tf`, `outputs.tf`, `versions.tf`, `*.tfvars`.
Reference: `TYPE.NAME.attr`, `data.T.N`, `var.x`, `local.x`, `module.m.out`.

TYPES · EXPRESSIONS

```
string number bool · list(T) set(T)
map(T) · object({...}) tuple({...})
· any, null.
```

`var.prod ? "m7g.large" : "t4g.mic
ro"`
`{for s in var.names : upper(s) if
s != ""}`
`{for k, v in var.tags : k => lowe
r(v)}`
`aws_instance.web[*].id #
splat`
`"${for n in var.l }${n},${endfo
r}"`
`type = object({
 name = string
 size = optional(number, 10)
})`
`dynamic "ingress" {
 for_each = var.ports
 content { from_port = ingress.v
alue }
}`

15 functions: `lookup` `merge` `concat` `flatten` `length` `contains` `coalesce` `try` `can` `format` `join` `splits` `cidrsubnet` `templatefile` `jsonencode` `yamldecode` `toset`. Also `provider::terraform::encode_tfvars`, `decode_tfvars`, `encode_expr`.

META-ARGUMENTS

	count	for_each
Keys	index [0]	string ["a"]
Input	number	map / set(string)
Remove middle	shifts & replaces	only that one

- Use `count = var.on ? 1 : 0` for toggles; `for_each` for collections. Keys must be known at plan time.
- `depends_on` only for hidden dependencies.
- `provider = aws.us` on a resource; `providers = { aws = aws.us }` on a module.
- lifecycle:** `create_before_destroy`, `prevent_destroy`, `ignore_changes`, `replace_triggered_by`, `precondition`, `postcondition`.

VARIABLE PRECEDENCE

Lowest → highest (later wins, no merge):
`default` → `TF_VAR_x` → `terraform.tfvars` → `terraform.tfvars.json` → `*.auto.tfvars(.json)` (lexical) → `-var` / `-var-file` (in order; HCP workspace vars here). No value + `-input=false` = error.

VALIDATION

```
variable "env" {
  type = string
  validation {
    condition = contains(["dev", "prod"], var.env)
    error_message = "env must be dev or prod."
  }
  lifecycle {
    precondition { condition = ... , error_message = "..." }
    postcondition { condition = sel
f, ready, error_message = "..." }
  }
  check "site_up" {
    data "http" "home" { url = "htt
ps://example.com" }
    assert { condition = data.http.
home.status_code == 200
      error_message = "down"
    }
  }
}
```

validation/pre/post fail the run; check only warns.

STATE · BACKENDS

```
backend "s3" {
  bucket = "bookshelf-tfsta
te"
  key = "prod/terraform.
tfstate"
  region = "eu-west-1"
  encrypt = true
  use_lockfile = true # no Dyna
moDB
}
```

- Backends: `local`, `s3`, `azurerm`, `gcs`, `oci`, `pg`, `http`, `remote/cloud` block. No variables in backend: use `-backend-config`.
- Locking on every write; **force-unlock** only when sure.
- State holds secrets in plain text: encrypt, restrict, version the bucket.
- Remote state:** `data "terraform_remote_state"` or (better) publish outputs to SSM / tags and read with data sources.
- Workspaces = several states per config; prefer a directory per environment for prod isolation.

MODULES

- source:** `./modules/net`, registry `ns/name/provider (+ version)`, `git::https://...?ref=v1.2.0`, `github.com/org/repo, s3::, gcs::`, HTTP archive.
- Version only registry modules; pin git with a tag `?ref=`. SemVer tags `v1.2.0`.
- Standard layout: `main.tf`, `variables.tf`, `outputs.tf`, `versions.tf`, `README.md`, `modules/`, `examples/`, `tests/`.
- No `provider` blocks in reusable modules; declare **required_providers** with `>=` minimums.

REFACTORING · IMPORT

```
moved {
  from = aws_s3_bucket.oid
  to = module.store.aws_s3_buck
et.this
}
removed {
  from = aws_s3_bucket.legacy
  lifecycle { destroy = false }
}
import {
  to = aws_s3_bucket.covers
  id = "bookshelf-covers"
}
```

`terraform plan -generate-config-o
ut-gen.tf`

- `import` supports `for_each`, `identity` (1.12) and blocks in modules (1.16).
- `terraform query` (1.14): `list` blocks in `*.tfquery.hcl` find existing resources; `-generate-config-out` writes `import` + `config`.

SECRETS

Kind	In plan/state?
<code>sensitive = true</code>	redacted in output, stored in state
<code>ephemeral</code>	never persisted
<code>var/output/resourc</code> <code>e</code> (1.10)	
<code>write-only *wo + *wo_version</code>	sent to provider, never stored
<code>ephemeral "aws_secretsmanager_sec</code> <code>ret_version" "db" {</code> <code>secret_id = "bookshelf/db"</code> <code>}</code>	
<code>resource "aws_db_instance" "db" {</code> <code>password_wo</code> <code>aws_secretsmanager_secret_versi</code> <code>on.db.secret_string</code> <code>password_wo_version = 1</code> <code>}</code>	

TESTING

```
# tests/bucket.tf test.hcl
mock_provider "aws" {}
variables { env = "dev" }
run "name_has_prefix" {
  command = plan
  assert {
    condition = startswith(aw
s_s3_bucket.covers.bucket, "books
help")
    error_message = "bad bucket n
ame"
  }
}
run "rejects_bad_env" {
  command = plan
  variables { env = "qa" }
  expect_failures = [var.env]
}
```

`terraform test -junit-xml=out.xml`. Also `override_resource`, `override_data`, `override_module`; `Terratest` (Go) for real infra; `LocalStack`.

QUALITY · POLICY

- TFLint** (+ `aws/azurerm/google` rulesets) · **Checkov** · **Trivy** `trivy config` · **terraform-docs** · `pre-commit` · `tenv` for versions.
- Sentinel** (HCP / Enterprise, `tfplan/v2`; advisory · `soft` · hard mandatory).
- OPA/Rego:** HCP policy sets or `Conftest` on `show -json`.
- Terraform Policy:** `tfpolicy`, `-policies` flag (youngest).

CI · OIDC

```
permissions: { id-token: write, c
ontents: read }
env: { TF_IN_AUTOMATION: "true" }
steps:
  - uses: actions/checkout@v7
  - uses: hashicorp/setup-terrafo
rm@v4
    with: { terraform_version: 1.
17.0 }
  - uses: aws-actions/configure-a
ws-credentials@v6
    with: { role-to-assume: ${{ v
ars.ROLE_ARN }},
          aws-region: eu-west-1
        }
  - run: terraform init -input=fal
se
  - run: terraform plan -out=tfpl
an -input=false
```

- Short-lived creds via **OIDC** (AWS role, Azure federated credential, Google WIF). No stored keys.
- PR: `fmt`, `validate`, `TFLint`, `test`, `plan` comment. Merge: apply saved plan behind a protected environment.
- Scheduled plan -refresh-only -detailed-exitcode** for drift. `Atlantis` / `GitLab` alternatives.

HCP TERRAFORM · STACKS

- HCP Terraform** (ex Terraform Cloud): remote runs, state, VCS/CLI/API workspaces, projects, variable sets, policy, private registry, dynamic credentials. `cloud { organization, workspaces }` block. Enterprise = self-hosted.
- Stacks:** `*.tfcomponent.hcl` components + `*.tfdeploy.hcl` deployments, one config many environments/regions; `terraform stacks` CLI (1.13).

TERRAFORM VS OPENTOFU

	Terraform 1.17	OpenTofu 1.13
Licence	BSL 1.1 (IBM/Hashi Corp)	MPL-2.0, Linux Foundation / CNCF
CLI · files	<code>terraform, .tf</code>	<code>tofu, .tf + .tofu</code>
Registry	<code>registry.terraform.io</code>	<code>registry.opentofu.org</code> , <code>OCI</code>
Only here	Stacks, HCP, actions, query, Terraform Policy	state/plan encryption, early eval, provider <code>for_each</code> , <code>-exclude</code> , <code>enabled</code>

Same language, providers and TF_ env vars; forked from 1.5.x.

TROUBLESHOOTING

Symptom	Cause → fix
Error acquiring the state lock	other/crashed run → wait; force-unlock ID
Inconsistent dependency lock file	constraints changed → init -upgrade ; providers lock all platforms
401 / 403	wrong creds/role → check with cloud CLI, TF_LOG_PROVIDER=DEBUG
Invalid for_each / count argument	keys unknown until apply → key by known values
Cycle	mutual references → break link; graph -draw-cycles
Instance cannot be destroyed	prevent_destroy → remove deliberately
Unexpected replace	force-new arg / rename → read plan; moved block
Wants to create existing	made outside TF → import block
Diff every run	<code>drift / timestamp()</code> → <code>-refresh-only</code> , <code>planteimestamp()</code> , <code>ignore_changes</code>
Inconsistent result after apply	provider bug → upgrade, report with log
Backend init error	backend changed → -reconfigure / -migrate-state
Variables not allowed (backend)	-backend-config partial config

CHANGED SINCE OLDER TUTORIALS

- Terraform Cloud → **HCP Terraform** (2024); legacy Free plan ended 2026-03-31.
- `S3 + DynamoDB` locking → `use_lockfile = true` (1.10, GA 1.11); `DynamoDB` args deprecated.
- `taint` → `-replace=ADDR`; `refresh` → `-refresh-only`.
- `terraform import` CLI → **import** blocks (1.5) + `query` (1.14).
- `template_file` → `templatefile()`; `-state` flag deprecated (1.10).
- Secrets in state → `ephemeral` (1.10) / `write-only` (1.11).
- `CDKTF` archived 2025-12-10 (`cdktf synth --hcl`).
- `tfsec` → **Trivy**.
- `MPL` → `BSL` (2023) → **OpenTofu** fork (1.6, 2024).
- AWS** → **5.0** → **6.x**; checkout v7, `setup-terraform v4`.