Gorgo Tasting OpenTofu and Terraform

September 08, 2026

Appendix A

Best Practices, Recommendations, and Common Errors

This appendix is the booklet's reference card. The chapters introduce these ideas where they come up; this is where they are collected, with the reasoning kept short and a pointer to the documentation for each item. Everything here applies to OpenTofu and Terraform alike unless a line says otherwise, and the two projects' documentation sites cover the same ground in the same words most of the time [1], [2].

Best practices

Organizing configuration

- **One directory per root module, one state per root module.** A root module is a unit of `apply`. Things with different lifecycles (a cluster and what runs on it, infrastructure and application) go in different root modules, as Chapters 4 and 5 do [3], [4].
- **Use the conventional file names.** `versions.tf` (the `terraform` block), `providers.tf`, `variables.tf`, `outputs.tf`, `main.tf`, and more files named for what they hold (`network.tf`, `dns.tf`). OpenTofu does not care; the next reader does [4], [5].
- **Name resources for their role, not their type:** `web`, not `web_container`. The type is already in the address. Use underscores in names, never dashes, and keep names short: they appear in every plan [4], [6].
- **Keep modules small and single purpose,** with every variable described and every output named for what it is. A module is a function; write it like one [5].
- **Do not abstract too early.** Three similar resources are fine. The third copy of a whole configuration is the moment to write a module [7].
- **Give every variable a `type` and a `description`,** and a `validation` block when a wrong value would fail late or expensively [8].
- **Prefer references to `depends_on`.** A reference is a dependency the reader can see. Use `depends_on` only when nothing can be referenced, and comment why [9].

Versions and providers

- **Constrain the provider major version** with `~>` in `required_providers`, and set `required_version` to the oldest tool version you tested [10], [11].
- **Commit `.terraform.lock.hcl`.** It pins exact provider versions and checksums so that everyone gets the same plugins. When the team spans platforms, run `tofu providers lock -platform=linux_amd64 -platform=darwin_arm64` and commit the result, or `init` on the other platform fails with a checksum

error [12], [13]. (The examples in this booklet ignore the lock file for that reason; a real project should not.)
- **Upgrade providers deliberately** with `tofu init -upgrade`, read the changelog first, and commit the lock file change on its own [14].
- **Never commit `.terraform/`.** It is a cache [12].

State

- **Never commit state.** It holds secrets and it changes on every apply. `.gitignore` gets `*.tfstate`, `*.tfstate.*`, `.terraform/`, and `crash.log` before the first apply [15].
- **Use a remote backend with locking** as soon as two people or one CI job touch a configuration: S3 and compatible stores (Oracle Object Storage among them), Azure Blob, Google Cloud Storage, a Kubernetes secret, or an HTTP backend. Locking prevents two applies from interleaving; encryption at rest covers the secrets [16], [17], [18].
- **Never edit state by hand.** `tofu state list`, `show`, `mv`, `rm`, and `tofu import` cover what is needed, and `moved`, `removed`, and `import` blocks do the same declaratively and reviewably [19], [20], [21].
- **Back up state** before anything unusual, with `tofu state pull > backup.tfstate` [22].
- **One state per environment,** by directory or by backend key, not by `workspace` unless the differences are truly only variable values [23].

Planning and applying

- **Always read the plan.** The summary line first, then every `-`/`+`. A replacement of anything stateful is a stop-and-think moment [24].
- **Save the plan in automation:** `tofu plan -out=tfplan` then `tofu apply tfplan`. A saved plan applies exactly what was reviewed, and applying a stale one fails instead of surprising you [24], [25].
- **Run `fmt`, `validate`, and `plan` in CI on every pull request,** and apply from CI on merge, so that state changes are serialized and logged [26], [27].
- **Avoid `-target`.** It is for surgery, not routine. A configuration that needs `-target` to apply cleanly has a dependency problem to fix [24].
- **Do not `-auto-approve` anything that matters.** Interactively, read and type `yes`; in CI, apply a saved plan that was reviewed [25].
- **Treat `destroy` as a real command.** Protect stateful resources with `lifecycle { prevent_destroy = true }` and remove the protection only in the same change that intends the destruction [28], [29].

Secrets

- **Credentials come from the environment or the vendor's own config files,** never from `.tf` or committed `.tfvars` files. `TF_VAR_name` sets a variable from the environment [30], [31].
- **Mark secret variables and outputs `sensitive`,** and wrap values read from another configuration's outputs in `sensitive()` [8], [32].
- **Remember that state holds the plain value regardless.** Sensitivity is about the terminal and the logs; the backend's access control is about the state [15].
- **Generate secrets with the `random` provider or fetch them from a secret manager** rather than typing them, so that they are never in a shell history [33].

Changing infrastructure

- **Prefer immutable artifacts.** An image tagged with its content hash (Chapter 4) is a change OpenTofu can see; `latest` is not [34], [35].
- **Provisioners are a last resort.** They run only at creation, their failures leave resources tainted, and their commands are invisible to the plan. Prefer a provider, cloud-init, or a proper configuration management tool, and use `terraform_data` with `local-exec` for the small glue that has no better home [36], [37].

- **Make replacement chains explicit** with `replace_triggered_by` when the provider cannot see the dependency (a seed file and the volume it loads into) [29].
- **Add `precondition` and `postcondition` blocks** for assumptions that would otherwise fail late, and `check` blocks for "did it work" [38], [39].
- **Refactor with `moved` blocks,** not by destroying and recreating. Renaming a resource or moving it into a module is a `moved { from = ...; to = ... }` and a plan that says "no changes" [20].

Recommendations

Tools

- `tofu fmt -recursive` and `tofu validate` in a pre-commit hook [40], [41], [42].
- **tflint** for lint rules OpenTofu does not enforce (unused variables, deprecated arguments, provider-specific checks) [43].
- **trivy** or **checkov** for security scanning of configurations (open security groups, unencrypted storage) [44], [45].
- **terraform-docs** to generate the variables and outputs tables of a module's README from its files [46].
- **tenv** (or a similar version manager) to install and switch between OpenTofu and Terraform versions per project [47].
- An editor with the language server, which gives completion for every provider's arguments [48].

Working with OpenTofu

- `tofu console` to try an expression before putting it in a file [49].
- `tofu show` to read the state or a saved plan; `tofu show -json tfplan` for scripts [50].
- `tofu graph | dot -Tsvg > graph.svg` to see the dependency graph when the order surprises you [51].
- `tofu plan -refresh-only` to see drift (changes made outside OpenTofu) without planning any fixes, and `tofu apply -refresh-only` to accept it into the state [24], [52].
- `tofu apply -replace=ADDRESS` to force one resource to be recreated, instead of tainting or deleting things [24].
- `TF_LOG=DEBUG tofu plan` when a provider does something inexplicable; it prints every API call [53].
- `tofu test` with `.tftest.hcl` files for modules that other configurations depend on [54].
- `tofu providers schema -json` to see every argument a provider accepts when the documentation and reality disagree [55].

Staying compatible with both tools

Everything in this booklet works in both. Features to avoid, or to fence off, if a configuration must run under either [56]:

Only in OpenTofu	Only in Terraform
State encryption (`encryption` block) [57]	HCP Terraform and `cloud` block features [58]
Variables in `backend` and module `source` [59]	Terraform Stacks [60]
`for_each` on `provider` blocks [61]	`ephemeral` resources and write-only arguments [62], [63]
`-exclude` flag [61]	`terraform query` and list resources [64]
`.tofu` file extension [59]	Some newer built-in functions [2]
Provider mirrors from OCI registries [65], [66]	

Both have `check` blocks, `import`, `moved`, and `removed` blocks, `terraform_data`, provider-defined functions, and `test` [67]. The version numbers do not line up: OpenTofu started at 1.6 (equivalent to Terraform 1.5), so `required_version = ">= 1.6"` means something slightly different under each tool [56].

Common errors

The message, what it usually means, and what to do. Messages are abbreviated.

Setup

Message	Cause and fix
`Required plugins are not installed` or `Missing required provider`	Run `tofu init` [14].
`Inconsistent dependency lock file`	The lock file and the constraints disagree; `tofu init` if a constraint changed, `tofu init -upgrade` to move to newer versions [12], [14].
`Backend initialization required`	The backend settings changed; `tofu init -migrate-state` to move state, `-reconfigure` to point at existing state [14], [18].
`Failed to query available provider packages`	Wrong `source` address, a typo in the version, or no network. Check the registry page for the exact address [11].
`Unsupported OpenTofu Core version`	`required_version` excludes the version you are running; upgrade OpenTofu or loosen the constraint [68].
`Error acquiring the state lock`	Another apply is running, or one crashed and left the lock; wait, then `tofu force-unlock ID` only when you are sure it is stale [17], [69].

Configuration

Message	Cause and fix
`Unsupported argument` (with "Did you mean")	A misspelled argument, or an argument from a different provider version [41].
`Missing required argument`	The resource type needs it; the docs list which are required [70].
`Reference to undeclared resource` or `undeclared input variable`	A typo in a reference, or a missing `variable` block. Remember `var.`, `local.`, `data.`, `module.` prefixes [71].
`Invalid function argument` or `Invalid template interpolation value`	A type mismatch, often a number where a string is needed; use `tostring`, `tonumber`, or check the type in `tofu console` [72], [73].
`Invalid for_each argument ... known after apply`	`for_each` keys must be known at plan time; use static keys, derive them from configuration rather than from another resource's attributes, or split into two applies [74].
`Cycle:` followed by addresses	Resources reference each other in a loop; break it with a data source, a local, or by removing an unneeded `depends_on` [9], [75].
`Duplicate resource`	Two blocks with the same type and name, often in two files [70].
`Invalid value for variable`	A `validation` block rejected the value; the message is whatever the author wrote [8].
`Resource precondition failed`	A `precondition` was false; fix the assumption or the configuration [38].

Planning and applying

Message	Cause and fix
`Saved plan is stale`	State changed since the plan was saved; plan again [25].
`Objects have changed outside of OpenTofu`	Drift. Read what changed; `apply -refresh-only` to accept it, or a normal apply to revert it [24], [52].
`Provider produced inconsistent final plan`	A provider bug or a value that changed between plan and apply; re-run, then report it if it repeats [53].
`Instance cannot be destroyed`	`prevent_destroy` is doing its job; remove it deliberately if you mean it [29].
`local-exec provisioner error`	The command failed; the resource is tainted and the next apply recreates it. Fix the command [36].
`Check block assertion failed`	A warning: the apply succeeded but the check did not. Look at what the check tests [39].
`Provider configuration not present`	A resource in state belongs to a provider you removed from the configuration; add the provider back long enough to destroy it, or `tofu state rm` it [31].
`Error: ... already exists` or HTTP 409	The thing exists but not in state; `tofu import` it or an `import` block, or remove the stray one [21], [76].

Providers used in this booklet

Message	Cause and fix
`Bind for 0.0.0.0:8080 failed: port is already allocated`	Another process or container owns the port; change the variable [77].
`Conflict. The container name "/motd-db" is already in use`	A container OpenTofu does not know about; remove it or import it [76], [78].
`Cannot connect to the Docker daemon`	Docker is not running, or the socket needs your user in the `docker` group, or the `host` is wrong [79], [80].
`timed out waiting for the condition` (Kubernetes)	The rollout never became ready; `kubectl describe pod` and `kubectl logs` say why [81], [82].
`ImagePullBackOff`	The cluster cannot pull the image: not loaded into kind, not pushed to the registry, or no pull secret [34], [83].
`Unauthorized` (Kubernetes)	The kubeconfig context is wrong or expired [84].
`Out of host capacity` (Oracle)	No free tier machines in that availability domain; try another `ad_index` or region [85], [86].
`NotAuthorizedOrNotFound` (Oracle)	The compartment OCID is wrong, or the profile in `~/.oci/config` lacks permission, or the region differs from the resource's [87], [88].
`Authentication error (10000)` (Cloudflare)	The token is missing, wrong, or lacks `Zone.DNS: Edit` on that zone [89], [90].
`urn:ietf:params:acme:error:rateLimited`	Too many certificates for the name; wait, and use the staging server while testing [91], [92], [93].
`NXDOMAIN` during the DNS challenge	The TXT record has not propagated; the provider retries, and a low TTL (time to live) on the zone helps [94], [95].
`Permission denied (publickey)` over SSH	The key in `ssh_public_key` is not the one your agent offers, or cloud-init has not finished creating the user [96], [97].

[1] OpenTofu Project, "OpenTofu documentation." 2026. Available: https://opentofu.org/docs/

[2] HashiCorp, “Terraform documentation.” 2026. Available: https://developer.hashicorp.com/terraform/docs

[3] OpenTofu Project, “Module composition.” 2026. Available: https://opentofu.org/docs/language/modules/develop/composition/

[4] HashiCorp, “Terraform style guide.” 2026. Available: https://developer.hashicorp.com/terraform/language/style

[5] OpenTofu Project, “Standard module structure.” 2026. Available: https://opentofu.org/docs/language/modules/develop/structure/

[6] OpenTofu Project, “Style conventions.” 2026. Available: https://opentofu.org/docs/language/syntax/style/

[7] OpenTofu Project, “Creating modules.” 2026. Available: https://opentofu.org/docs/language/modules/develop/

[8] OpenTofu Project, “Input variables.” 2026. Available: https://opentofu.org/docs/language/values/variables/

[9] OpenTofu Project, “The depends_on meta-argument.” 2026. Available: https://opentofu.org/docs/language/meta-arguments/depends_on/

[10] OpenTofu Project, “Version constraints.” 2026. Available: https://opentofu.org/docs/language/expressions/version-constraints/

[11] OpenTofu Project, “Provider requirements.” 2026. Available: https://opentofu.org/docs/language/providers/requirements/

[12] OpenTofu Project, “Dependency lock file.” 2026. Available: https://opentofu.org/docs/language/files/dependency-lock/

[13] OpenTofu Project, “Command: Providers lock.” 2026. Available: https://opentofu.org/docs/cli/commands/providers/lock/

[14] OpenTofu Project, “Command: init.” 2026. Available: https://opentofu.org/docs/cli/commands/init/

[15] OpenTofu Project, “Sensitive data in state.” 2026. Available: https://opentofu.org/docs/language/state/sensitive-data/

[16] OpenTofu Project, “Remote state.” 2026. Available: https://opentofu.org/docs/language/state/remote/

[17] OpenTofu Project, “State locking.” 2026. Available: https://opentofu.org/docs/language/state/locking/

[18] OpenTofu Project, “Backend configuration.” 2026. Available: https://opentofu.org/docs/language/settings/backends/configuration/

[19] OpenTofu Project, “Command: state.” 2026. Available: https://opentofu.org/docs/cli/commands/state/

[20] OpenTofu Project, “Refactoring.” 2026. Available: https://opentofu.org/docs/language/modules/develop/refactoring/

[21] OpenTofu Project, “Import.” 2026. Available: https://opentofu.org/docs/language/import/

[22] OpenTofu Project, “Command: State pull.” 2026. Available: https://opentofu.org/docs/cli/commands/state/pull/

[23] OpenTofu Project, “Workspaces.” 2026. Available: https://opentofu.org/docs/language/state/workspaces/

[24] OpenTofu Project, “Command: plan.” 2026. Available: https://opentofu.org/docs/cli/commands/plan/

[25] OpenTofu Project, “Command: apply.” 2026. Available: https://opentofu.org/docs/cli/commands/apply/

[26] OpenTofu Project, "The core OpenTofu workflow." 2026. Available: https://opentofu.org/docs/intro/core-workflow/

[27] HashiCorp, "Running Terraform in automation." 2026. Available: https://developer.hashicorp.com/terraform/tutorials/automation/automate-terraform

[28] OpenTofu Project, "Command: destroy." 2026. Available: https://opentofu.org/docs/cli/commands/destroy/

[29] OpenTofu Project, "The lifecycle meta-argument." 2026. Available: https://opentofu.org/docs/language/meta-arguments/lifecycle/

[30] OpenTofu Project, "Environment variables." 2026. Available: https://opentofu.org/docs/cli/config/environment-variables/

[31] OpenTofu Project, "Provider configuration." 2026. Available: https://opentofu.org/docs/language/providers/configuration/

[32] OpenTofu Project, "Sensitive function." 2026. Available: https://opentofu.org/docs/language/functions/sensitive/

[33] HashiCorp, "Terraform provider for random: random_password." 2026. Available: https://github.com/hashicorp/terraform-provider-random/blob/main/docs/resources/password.md

[34] The Kubernetes Authors, "Images." 2026. Available: https://kubernetes.io/docs/concepts/containers/images/

[35] The Kubernetes Authors, "Configuration best practices." 2026. Available: https://kubernetes.io/docs/concepts/configuration/overview/

[36] OpenTofu Project, "Provisioners." 2026. Available: https://opentofu.org/docs/language/resources/provisioners/syntax/

[37] HashiCorp, "The terraform_data managed resource type." 2026. Available: https://developer.hashicorp.com/terraform/language/resources/terraform-data

[38] OpenTofu Project, "Custom condition checks." 2026. Available: https://opentofu.org/docs/language/expressions/custom-conditions/

[39] OpenTofu Project, "Checks with assertions." 2026. Available: https://opentofu.org/docs/language/checks/

[40] OpenTofu Project, "Command: fmt." 2026. Available: https://opentofu.org/docs/cli/commands/fmt/

[41] OpenTofu Project, "Command: validate." 2026. Available: https://opentofu.org/docs/cli/commands/validate/

[42] A. Babenko, "Pre-commit-terraform." 2026. Available: https://github.com/antonbabenko/pre-commit-terraform

[43] TFLint Contributors, "TFLint: A pluggable Terraform linter." 2026. Available: https://github.com/terraform-linters/tflint

[44] Aqua Security, "Trivy." 2026. Available: https://trivy.dev/

[45] Prisma Cloud, "Checkov." 2026. Available: https://www.checkov.io/

[46] terraform-docs Contributors, "Terraform-docs." 2026. Available: https://terraform-docs.io/

[47] tofuutils, "Tenv: OpenTofu and Terraform version manager." 2026. Available: https://github.com/tofuutils/tenv

[48] OpenTofu Project, "OpenTofu language server." 2026. Available: https://github.com/opentofu/tofu-ls

[49] OpenTofu Project, "Command: console." 2026. Available: https://opentofu.org/docs/cli/commands/console/

[50] OpenTofu Project, "Command: show." 2026. Available: https://opentofu.org/docs/cli/commands/show/

[51] OpenTofu Project, "Command: graph." 2026. Available: https://opentofu.org/docs/cli/commands/graph/

[52] OpenTofu Project, "Command: refresh." 2026. Available: https://opentofu.org/docs/cli/commands/refresh/

[53] OpenTofu Project, "Debugging OpenTofu." 2026. Available: https://opentofu.org/docs/internals/debugging/

[54] OpenTofu Project, "Command: test." 2026. Available: https://opentofu.org/docs/cli/commands/test/

[55] OpenTofu Project, "Command: Providers schema." 2026. Available: https://opentofu.org/docs/cli/commands/providers/schema/

[56] OpenTofu Project, "Migrating to OpenTofu from Terraform." 2026. Available: https://opentofu.org/docs/intro/migration/

[57] OpenTofu Project, "State and plan encryption." 2026. Available: https://opentofu.org/docs/language/state/encryption/

[58] HashiCorp, "HCP Terraform documentation." 2026. Available: https://developer.hashicorp.com/terraform/cloud-docs

[59] OpenTofu Project, "What's new in OpenTofu 1.8." 2026. Available: https://opentofu.org/docs/v1.8/intro/whats-new/

[60] HashiCorp, "Terraform stacks." 2026. Available: https://developer.hashicorp.com/terraform/language/stacks

[61] OpenTofu Project, "What's new in OpenTofu 1.9." 2026. Available: https://opentofu.org/docs/v1.9/intro/whats-new/

[62] HashiCorp, "Ephemeral resources." 2026. Available: https://developer.hashicorp.com/terraform/language/resources/ephemeral

[63] HashiCorp, "Write-only arguments." 2026. Available: https://developer.hashicorp.com/terraform/language/resources/ephemeral/write-only

[64] HashiCorp, "Command: query." 2026. Available: https://developer.hashicorp.com/terraform/cli/commands/query

[65] OpenTofu Project, "What's new in OpenTofu 1.10." 2026. Available: https://opentofu.org/docs/v1.10/intro/whats-new/

[66] OpenTofu Project, "OCI registries." 2026. Available: https://opentofu.org/docs/cli/oci_registries/

[67] OpenTofu Project, "What's new in OpenTofu 1.7." 2026. Available: https://opentofu.org/docs/v1.7/intro/whats-new/

[68] OpenTofu Project, "OpenTofu settings." 2026. Available: https://opentofu.org/docs/language/settings/

[69] OpenTofu Project, "Command: Force-unlock." 2026. Available: https://opentofu.org/docs/cli/commands/force-unlock/

[70] OpenTofu Project, "Resource blocks." 2026. Available: https://opentofu.org/docs/language/resources/syntax/

[71] OpenTofu Project, "References to named values." 2026. Available: https://opentofu.org/docs/language/expressions/references/

[72] OpenTofu Project, "Types and values." 2026. Available: https://opentofu.org/docs/language/expressions/types/

[73] OpenTofu Project, "Tostring function." 2026. Available: https://opentofu.org/docs/language/functions/tostring/

[74] OpenTofu Project, "The for_each meta-argument." 2026. Available: https://opentofu.org/docs/language/meta-arguments/for_each/

[75] OpenTofu Project, "Resource graph." 2026. Available: https://opentofu.org/docs/internals/graph/

[76] OpenTofu Project, "Command: import." 2026. Available: https://opentofu.org/docs/cli/import/

[77] Docker Inc., "Networking overview." 2026. Available: https://docs.docker.com/engine/network/

[78] Docker Inc., "Docker container run." 2026. Available: https://docs.docker.com/reference/cli/docker/container/run/

[79] Docker Inc., "Linux post-installation steps for docker engine." 2026. Available: https://docs.docker.com/engine/install/linux-postinstall/

[80] kreuzwerker, "Terraform provider for docker: docker_container." 2026. Available: https://github.com/kreuzwerker/terraform-provider-docker/blob/master/docs/resources/container.md

[81] HashiCorp, "Terraform provider for kubernetes: kubernetes_deployment_v1." 2026. Available: https://github.com/hashicorp/terraform-provider-kubernetes/blob/main/docs/resources/deployment_v1.md

[82] The Kubernetes Authors, "Debug pods." 2026. Available: https://kubernetes.io/docs/tasks/debug/debug-application/debug-pods/

[83] The Kubernetes Authors, "Kind quick start." 2026. Available: https://kind.sigs.k8s.io/docs/user/quick-start/

[84] The Kubernetes Authors, "Organizing cluster access using kubeconfig files." 2026. Available: https://kubernetes.io/docs/concepts/configuration/organize-cluster-access-kubeconfig/

[85] Oracle, "Always free resources." 2026. Available: https://docs.oracle.com/en-us/iaas/Content/FreeTier/freetier_topic-Always_Free_Resources.htm

[86] Oracle, "Oracle cloud free tier." 2026. Available: https://www.oracle.com/cloud/free/

[87] Oracle, "API errors." 2026. Available: https://docs.oracle.com/en-us/iaas/Content/API/References/apierrors.htm

[88] Oracle, "Terraform provider for oracle cloud infrastructure." 2026. Available: https://docs.oracle.com/en-us/iaas/Content/dev/terraform/home.htm

[89] Cloudflare, "Create API token." 2026. Available: https://developers.cloudflare.com/fundamentals/api/get-started/create-token/

[90] Cloudflare, "Terraform provider for cloudflare." 2026. Available: https://github.com/cloudflare/terraform-provider-cloudflare/blob/master/docs/index.md

[91] Internet Security Research Group, "Rate limits." 2026. Available: https://letsencrypt.org/docs/rate-limits/

[92] Internet Security Research Group, "Staging environment." 2026. Available: https://letsencrypt.org/docs/staging-environment/

[93] Internet Security Research Group, "Let's encrypt documentation." 2026. Available: https://letsencrypt.org/docs/

[94] C. Marchesi, "Terraform provider for ACME: acme_certificate." 2026. Available: https://github.com/vancluever/terraform-provider-acme/blob/main/docs/resources/certificate.md

[95] Internet Security Research Group, "Challenge types." 2026. Available: https://letsencrypt.org/docs/challenge-types/

[96] Oracle, "Connecting to your linux instance using SSH." 2026. Available: https://docs.oracle.com/en-us/iaas/Content/Compute/Tasks/accessinginstance.htm

[97] Canonical, "Cloud-init documentation." 2026. Available: https://cloudinit.readthedocs.io/en/latest/