EN

IaC Guide

Terraform, Ansible, GitOps for SMBs

Done with click-ops — full IaC pipeline, BOTUM real case

Terraform · Ansible · GitOps

Mars 2026

Cloud Journey Series — Post B14

Your cloud infrastructure was built by hand. A click here, an Azure portal session there, a VM created in the AWS console on a Friday night under pressure. Six months later, nobody knows what's running, why, or how to reproduce it. That's "click-ops" — and it's the leading source of technical debt in the IT organizations we work with at BOTUM.

Infrastructure as Code (IaC) is the structural answer. Not a trend, not a luxury — a prerequisite for any SMB that wants to master its cloud infrastructure in 2025.

Real IaC: Declaring Desired State

The fundamental distinction of IaC is philosophical: you declare what you want, not how to get there. The declarative approach is idempotent: applying the same code twice gives the same result, with no side effects. Your infrastructure becomes versioned code in Git.

Terraform: Cloud Provisioning

Terraform (HashiCorp, open-source) is the de facto standard for provisioning cloud resources: VMs, VNets, databases, load balancers, DNS, IAM. It talks to 3,000+ provider APIs.

```
# main.tf
terraform {
  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = "~> 3.90"
    }
  }
  backend "azurerm" {
    resource_group_name  = "rg-tfstate"
    storage_account_name = "botumtfstate"
    container_name       = "tfstate"
    key                  = "prod.terraform.tfstate"
  }
}

resource "azurerm_resource_group" "main" {
  name     = "rg-botum-prod"
  location = "canadacentral"
}

resource "azurerm_linux_virtual_machine" "app" {
  name                = "vm-app-prod-01"
  resource_group_name = azurerm_resource_group.main.name
  size                = "Standard_B2s"
  admin_username      = "azureuser"
  os_disk {
    storage_account_type = "Premium_LRS"
    disk_size_gb         = 128
  }
}
```

Key points: State file must be in a remote encrypted backend (never local, never in Git). Use modules for reusability. Always run terraform plan before apply.

Ansible: Configuration and Deployment

Terraform creates infrastructure. Ansible configures it. Once your Azure VM is created, Ansible installs Nginx, copies configs, creates user accounts — idempotently via SSH, no agent required.

```yaml
# playbooks/web-server.yml
- name: Configure web server
  hosts: web_servers
  become: yes
  tasks:
    - name: Install nginx
      ansible.builtin.apt:
        name: nginx
        state: present
    - name: Deploy nginx config
      ansible.builtin.template:
        src: templates/nginx.conf.j2
        dest: /etc/nginx/sites-available/botum
      notify: Reload nginx
    - name: Create app user
      ansible.builtin.user:
        name: botum
        system: yes
  handlers:
    - name: Reload nginx
      ansible.builtin.service:
        name: nginx
        state: reloaded
```

GitOps: Git as Single Source of Truth

Every infrastructure change goes through a Pull Request — an auditable change ticket with discussion, approval, and history. The GitHub Actions pipeline: auto plan on PRs, auto apply on merge with approval gate.

```
# .github/workflows/terraform.yml
on:
  pull_request:
    paths: ['terraform/**']
  push:
    branches: [main]
    paths: ['terraform/**']

jobs:
  plan:
    if: github.event_name == 'pull_request'
    steps:
      - uses: hashicorp/setup-terraform@v3
      - run: terraform init && terraform plan
        env:
          ARM_CLIENT_ID: ${{ secrets.ARM_CLIENT_ID }}
          ARM_CLIENT_SECRET: ${{ secrets.ARM_CLIENT_SECRET }}

  apply:
    if: github.ref == 'refs/heads/main'
    environment: production  # Manual approval required
    steps:
      - run: terraform apply -auto-approve
```

Terraform vs Ansible: Who Does What

Criteria	Terraform	Ansible
Main role	Provisioning (create/destroy)	Configuration (install, deploy)
Target	Cloud APIs (Azure, AWS, DNS)	Existing servers (OS, packages, files)
Approach	Declarative (HCL)	Declarative + YAML
State	State file (explicit mgmt)	Stateless (read from server)
Idempotence	Yes (native)	Yes (if well-written)
Learning curve	Medium (HCL, state)	Low (readable YAML)
SMB use case	VMs, VNets, DBs, LBs in cloud	Nginx, Postgres, app deployment

Simple rule: Terraform for everything in the cloud portal. Ansible for everything done via SSH on a server.

For SMBs: Start Simple

A single repo structure is enough to get started. The value of IaC explodes when you can create staging in 10 minutes from proven prod modules.

```
infra-repo/
├── terraform/
│   ├── environments/
│   │   ├── dev/
│   │   ├── staging/
│   │   └── prod/
│   └── modules/
│       ├── vm/
│       └── network/
├── ansible/
│   ├── inventory/
│   ├── playbooks/
│   └── roles/
└── .github/
    └── workflows/
        ├── terraform.yml
        └── ansible.yml
```

Pitfalls That Hurt

- State file local or in Git: contains secrets in plain text. Encrypted remote backend mandatory (Azure Blob CMK, S3 SSE).
- Hardcoded secrets in HCL/YAML: inject from Azure Key Vault, AWS Secrets Manager, or HashiCorp Vault via CI/CD.
- Unpinned Terraform versions: without required_version, an auto-update can break your pipeline overnight.
- No environment separation: a single state file for dev/staging/prod is a bomb. Separate directories and backends are mandatory.

BOTUM Real Case: 3 Environments, 10-Minute Deployment

B2B SaaS client, 35 employees, Azure stack. Initial situation: hand-built infrastructure, no up-to-date docs, new DevOps spent 3 weeks "understanding what's running". Post-incident recovery: 4 hours minimum.

Metric	Before	After IaC
Create staging environment	4 hours	**11 minutes**
Onboard new DevOps	3 weeks	**2 days**
Incidents from manual config	Regular	**0**

Change audit trail	None	**Full Git**

Conclusion

IaC is no longer an "advanced" practice. It's the minimum for: reproducibility, traceability, resilience, compliance. Start simple: one repo, Terraform for cloud, Ansible for VMs, GitHub Actions for the pipeline.

Go Further with BOTUM

Implementing IaC in your organization? BOTUM teams guide you from A to Z.

Discuss your project →

Complete PDF Guide

Download this IaC guide as a PDF.

Download Guide (PDF)