CloudForge Terraform Integration Guide

Document ID: PRD-CF-035 **Last Updated:** 2024-03-01 **Owner:** CloudForge Product Team **Classification:** Public

---

Overview

CloudForge provides native Terraform integration, allowing you to manage CloudForge resources using HashiCorp Terraform. This guide covers provider setup, resource management, and best practices.

Provider Installation

Requirements

- Terraform 1.0 or later
- CloudForge account with API access
- API token with appropriate permissions

Provider Configuration

```
terraform {
  required_providers {
    cloudforge = {
      source  = "novatech/cloudforge"
      version = "~> 2.0"
    }
  }
}

provider "cloudforge" {
  api_token = var.cloudforge_token  # Or use CLOUDFORGE_API_TOKEN env var
  org_id    = var.cloudforge_org_id
}
```

Authentication

Option 1: Environment Variable (Recommended)

```
export CLOUDFORGE_API_TOKEN="cft_xxxxxxxxxxxxx"
export CLOUDFORGE_ORG_ID="org_xxxxxxxxxxxxx"
```

Option 2: Provider Configuration

```
provider "cloudforge" {
  api_token = "cft_xxxxxxxxxxxxx"
  org_id    = "org_xxxxxxxxxxxxx"
}
```

Option 3: Variables File

```
# terraform.tfvars
cloudforge_token  = "cft_xxxxxxxxxxxxx"
cloudforge_org_id = "org_xxxxxxxxxxxxx"
```

Resources

Environments

```
resource "cloudforge_environment" "production" {
  name        = "production"
  description = "Production environment"
  type        = "production"

  settings {
    region           = "us-west-2"
    auto_scaling     = true
    monitoring_level = "detailed"
  }

  tags = {
    team        = "platform"
    cost_center = "engineering"
  }
}
```

```
resource "cloudforge_environment" "staging" {
  name        = "staging"
  description = "Staging environment"
  type        = "staging"

  settings {
    region = "us-west-2"
  }
}
```

Compute Resources

```
resource "cloudforge_instance" "web_server" {
  environment_id = cloudforge_environment.production.id
  name           = "web-server"

  instance_type = "m5.large"
  count         = 3

  network {
    vpc_id    = cloudforge_vpc.main.id
    subnet_id = cloudforge_subnet.private.id
  }

  scaling {
    min_instances = 2
    max_instances = 10
    target_cpu    = 70
  }

  health_check {
    path     = "/health"
    port     = 8080
    interval = 30
  }

  tags = {
    role = "web"
  }
}
```

Databases

```
resource "cloudforge_database" "main" {
  environment_id = cloudforge_environment.production.id
  name           = "main-db"

  engine         = "postgresql"
  engine_version = "15"
  instance_class = "db.r5.large"

  storage {
    allocated     = 100
    max_allocated = 500
    type          = "gp3"
    iops          = 3000
  }

  high_availability {
    enabled     = true
    multi_az    = true
    read_replicas = 2
  }

  backup {
    retention_period = 30
    window          = "03:00-04:00"
  }

  credentials {
    username = "admin"
    password = var.db_password  # Use secrets management
  }
}
```

Load Balancers

```
resource "cloudforge_load_balancer" "main" {
  environment_id = cloudforge_environment.production.id
  name           = "main-lb"
  type           = "application"

  listeners {
    port     = 443
    protocol = "HTTPS"

    default_action {
      type             = "forward"
      target_group_id  = cloudforge_target_group.web.id
    }
    ssl_certificate_id = cloudforge_certificate.main.id
  }

  listeners {
    port     = 80
    protocol = "HTTP"

    default_action {
      type        = "redirect"
      redirect_to = "https"
    }
  }

  health_check {
    path                = "/health"
    healthy_threshold   = 2
    unhealthy_threshold = 3
    interval            = 30
  }
}
```

Networking

```
resource "cloudforge_vpc" "main" {
  environment_id = cloudforge_environment.production.id
  name           = "main-vpc"
  cidr_block     = "10.0.0.0/16"

  enable_dns_hostnames = true
  enable_dns_support   = true
}

resource "cloudforge_subnet" "public" {
  vpc_id            = cloudforge_vpc.main.id
  name              = "public-subnet"
  cidr_block        = "10.0.1.0/24"
  availability_zone = "us-west-2a"
  public            = true
}

resource "cloudforge_subnet" "private" {
  vpc_id            = cloudforge_vpc.main.id
  name              = "private-subnet"
  cidr_block        = "10.0.2.0/24"
  availability_zone = "us-west-2a"
  public            = false
}

resource "cloudforge_security_group" "web" {
  vpc_id = cloudforge_vpc.main.id
  name   = "web-sg"

  ingress {
    from_port   = 443
    to_port     = 443
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
}
```

Data Sources

Existing Resources

```
data "cloudforge_environment" "existing" {
  name = "production"
}

data "cloudforge_vpc" "existing" {
  environment_id = data.cloudforge_environment.existing.id
  name           = "main-vpc"
}
```

Available Instance Types

```
data "cloudforge_instance_types" "available" {
  filter {
    min_cpu    = 4
    min_memory = 8192
  }
}
```

Regions and Availability Zones

```
data "cloudforge_regions" "all" {}

data "cloudforge_availability_zones" "current" {
  region = "us-west-2"
}
```

Import Existing Resources

Import Command

```
# Import existing environment
terraform import cloudforge_environment.production env_xxxxxxxxxxxxx

# Import existing instance
terraform import cloudforge_instance.web_server inst_xxxxxxxxxxxxx

# Import existing database
terraform import cloudforge_database.main db_xxxxxxxxxxxxx
```

Import Block (Terraform 1.5+)

```
import {
  to = cloudforge_environment.production
  id = "env_xxxxxxxxxxxxx"
}

import {
  to = cloudforge_database.main
  id = "db_xxxxxxxxxxxxx"
}
```

Bulk Import

Use CloudForge CLI to generate import configurations:

```
cloudforge terraform import --environment production --output imports.tf
```

State Management

Remote State with CloudForge

```
terraform {
  backend "cloudforge" {
    organization = "my-org"
    workspace    = "production"
  }
}
```

State Locking

CloudForge backend supports state locking: - Automatic lock on plan/apply - Lock timeout: 5 minutes - Force unlock available (use with caution)

```
# Force unlock if needed
terraform force-unlock LOCK_ID
```

Modules

Official Modules

NovaTech provides official modules:

```
module "web_app" {
  source  = "novatech/web-app/cloudforge"
  version = "1.0.0"

  environment_id = cloudforge_environment.production.id
  name           = "my-web-app"
  instances = {
    count = 3
    type  = "m5.large"
  }

  database = {
    engine = "postgresql"
    size   = "db.r5.large"
  }

  load_balancer = {
    type = "application"
  }
}
```

Available Modules

Module	Description
`web-app`	Complete web application stack
`api-service`	API service with auto-scaling
`database`	Managed database with replicas
`kubernetes`	EKS/GKE cluster
`networking`	VPC, subnets, security groups

Best Practices

Directory Structure

```
infrastructure/
   environments/
      production/
         main.tf
         variables.tf
         outputs.tf
         terraform.tfvars
      staging/
      development/
   modules/
      web-app/
      database/
      networking/
   shared/
      provider.tf
```

Variable Management

```
# variables.tf
variable "environment" {
  description = "Environment name"
  type        = string
}

variable "instance_count" {
  description = "Number of instances"
  type        = number
  default     = 2
}

variable "db_password" {
  description = "Database password"
  type        = string
  sensitive   = true
}
```

Outputs

```
# outputs.tf
output "environment_id" {
  description = "Environment ID"
  value       = cloudforge_environment.production.id
}

output "load_balancer_dns" {
  description = "Load balancer DNS name"
  value       = cloudforge_load_balancer.main.dns_name
}

output "database_endpoint" {
  description = "Database endpoint"
  value       = cloudforge_database.main.endpoint
  sensitive   = true
}
```

Secrets Management

Use SecureVault integration:

```
data "securevault_secret" "db_password" {
  path = "myapp/production/database/password"
}

resource "cloudforge_database" "main" {
  # ...
  credentials {
    username = "admin"
    password = data.securevault_secret.db_password.value
  }
}
```

CI/CD Integration

DevPipeline Integration

```yaml
# .devpipeline.yaml
stages:
  - name: terraform-plan
    steps:
      - name: Plan
        uses: terraform
        with:
          command: plan
          working_directory: infrastructure/production
          var_file: terraform.tfvars

  - name: terraform-apply
    when:
      branch: main
    approval: required
    steps:
      - name: Apply
        uses: terraform
        with:
          command: apply
          working_directory: infrastructure/production
          auto_approve: true
```

GitHub Actions

```yaml
- name: Terraform Apply
  uses: novatech/terraform-action@v1
  with:
    command: apply
    working_directory: infrastructure
  env:
    CLOUDFORGE_API_TOKEN: ${{ secrets.CLOUDFORGE_TOKEN }}
```

Troubleshooting

Common Errors

Authentication Failed:

```
Error: authentication failed: invalid API token
```

- Verify token is correct
- Check token hasn't expired
- Ensure token has required permissions

Resource Not Found:

```
Error: resource not found: env_xxxxxxxxxxxxx
```

- Verify resource ID is correct
- Check organization ID matches
- Ensure you have access to the resource

State Lock:

```
Error: Error acquiring the state lock
```

- Wait for other operations to complete
- Use `terraform force-unlock` if stuck (carefully)

API Reference

Provider Arguments

Argument	Description	Required
`api_token`	CloudForge API token	Yes
`org_id`	Organization ID	Yes
`api_endpoint`	Custom API endpoint	No

Full Resource Documentation

See: registry.terraform.io/providers/novatech/cloudforge/latest/docs

---

Related Documents: Getting Started (PRD-CF-001), API Reference (PRD-CF-010), Migration Guide (PRD-CF-020)