

# Microsoft Entra IDをTerraform で運用する

# 自己紹介



Tadayuki Onishi @FOLIO

 @kenchan0130

---

## 話すこと

- Terraformの概要
- TerraformへのDeep Dive
  - TerraformでMicrosoft Entra IDを運用する際に困ったことおよび、あれば対応策の紹介
- Microsoft Graph APIに対するぼやき

## 話さないこと


- Terraformの基本的な使い方
- Terraformの基本的な構文

# Terraformざっくり概要

# Terraformとは

宣言的なコードでインフラストラクチャを管理するツール




-  HashiCorp社がBSL<sup>[1]</sup>として提供
- 設定はHCL<sup>[2]</sup>という独自の言語で記述
- 設計、設定値に関して意図を残せる
- 変更内容がレビューができる
- (ある程度の) ドリフト検出<sup>[3]</sup>ができる

---

1. Business Source License 

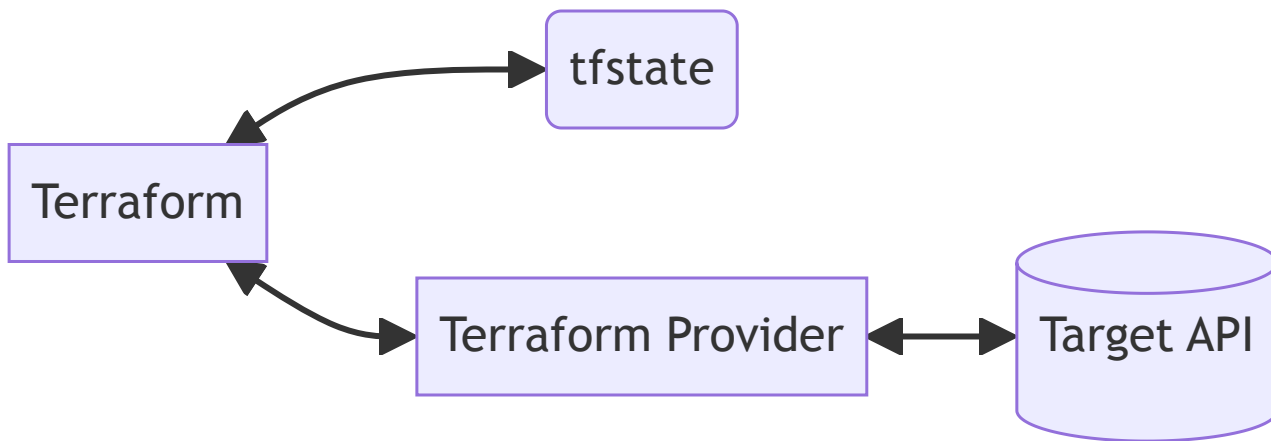
2. HashiCorp Configuration Language 

3. 管理しているリソースと実際のリソースの差分を検出する機能 

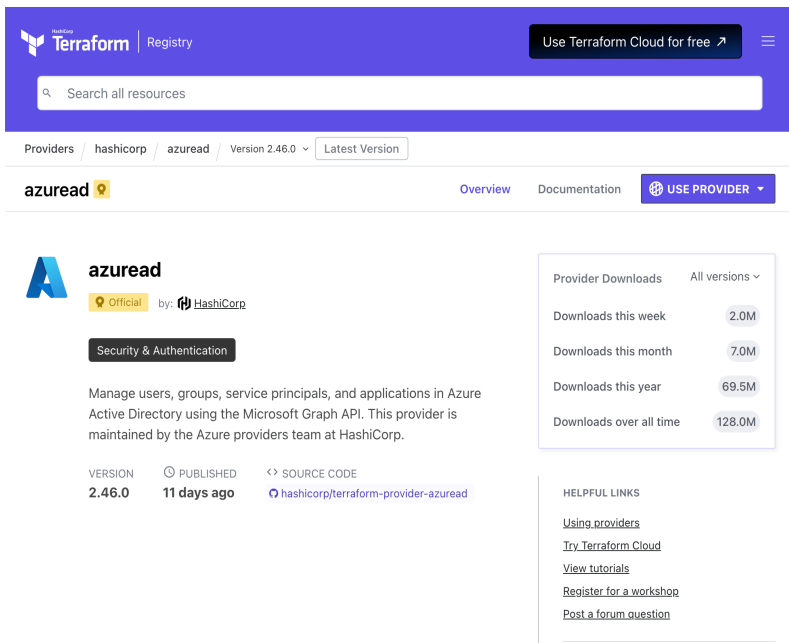
# Terraformの動作原理

Terraform Providerを介して各APIを呼び出す

tfstateには管理しているリソースの状態が記録される



# Terraform Provider for Azure Active Directory



The screenshot shows the Terraform Registry page for the `azuread` provider. The page is titled "azuread" and is marked as "Official" by HashiCorp. It features a "Security & Authentication" button and a description: "Manage users, groups, service principals, and applications in Azure Active Directory using the Microsoft Graph API. This provider is maintained by the Azure providers team at HashiCorp." The current version is 2.46.0, published 11 days ago. The page also displays download statistics for all versions and helpful links such as "Using providers", "Try Terraform Cloud", "View tutorials", "Register for a workshop", and "Post a forum question".

Provider Downloads	All versions
Downloads this week	2.0M
Downloads this month	7.0M
Downloads this year	69.5M
Downloads over all time	128.0M

- HashiCorp社の公式Provider
- 主にMicrosoft Graph APIを仲介する
- `manicminer/hamilton`がAPIクライアント<sup>[1]</sup>として使われている
  - manicminerさんはHashiCorp所属の方

1. 各ProviderはGo言語で記述されているため、Go言語のAPIクライアントが必要となる [👉](#)

# Terraformの基本的な操作

- `terraform init`
  - 依存ProviderやModuleのダウンロードなどを行う
- `terraform plan`
  - 構築される予定のリソースを模擬
- `terraform apply`
  - リソースの追加、変更や削除の実行

```
$ terraform init
Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/azurerm versions matching "2.46.0" ...
...
```

```
$ terraform plan
Terraform used the selected providers to generate the following execution plan:
+ create

Terraform will perform the following actions:
# azurerm_resource_group.kenchan0130 will be created
+ resource "azurerm_resource_group" "kenchan0130" {
+   name = "kenchan0130"
+   location = "Japan East"
+   tags = {
+     "about_me" = "kenchan0130"
+   }
}

Plan: 1 to add, 0 to change, 0 to destroy.
```

```
$ terraform apply
...
Apply: 1 to add, 0 to change, 0 to destroy.
```



# Microsoft Entra IDのリソース 管理の雰囲気をつかんでみる

# アプリケーション

アプリ登録に該当

```
resource "azuread_application" "google_workspace" {
  display_name = "Google Workspace"

  web {
    redirect_uris = [
      "https://www.google.com/a/example.com/acs"
    ]
  }

  lifecycle {
    ignore_changes = [
      identifier_uris,
    ]
  }
}

resource "azuread_application_identifier_uri" "google_workspace" {
  depends_on = [
    azuread_service_principal.google_workspace,
  ]

  application_id = azuread_application.google_workspace...
```

# サービsprincipal

エンタープライズアプリケーションに該当

```
resource "azuread_service_principal" "google_workspace" {
  client_id = azuread_application.google_workspace.client_id
  app_role_assignment_required = true
  preferred_single_sign_on_mode = "saml"

  saml_single_sign_on {
    relay_state = "https://www.google.com/a/example.com/..."
  }

  feature_tags {
    enterprise = true
    custom_single_sign_on = true
  }
}
```

# グループ


Microsoft Graph APIでは、Microsoft 365グループとセキュリティグループのみサポート  
[1]

## Microsoft 365グループ

```
resource "azuread_group" "bizdev_division" {
  display_name = "BizDev Division"
  mail_nickname = "bizdev-division-internal"
  types        = ["Unified"]
  mail_enabled = true
  owners = [
    azuread_user.kenchan0130.object_id,
  ]
}
```

## セキュリティグループ

```
resource "azuread_group" "bizdev_division" {
  display_name      = "BizDev Division"
  security_enabled = true
  owners = [
    azuread_user.kenchan0130.object_id,
  ]
}
```

- 
1. 「メールが有効なセキュリティグループ」、および「配布グループ」はExchange Online専用のオブジェクトのため、サポートされていない 

# 条件付きアクセス

条件付きアクセスポリシー、ネームドロケーションのみサポート

## 条件付きアクセスポリシー

```
resource "azuread_conditional_access_policy" "mfa" {
  display_name = "Enforce MFA"
  state        = "enabled"

  conditions {
    client_app_types = [
      "browser",
      "mobileAppsAndDesktopClients",
    ]

    applications {
      included_applications = [
        "All",
      ]
    }
  }

  users {
    included_users = [
```

## ネームドロケーション

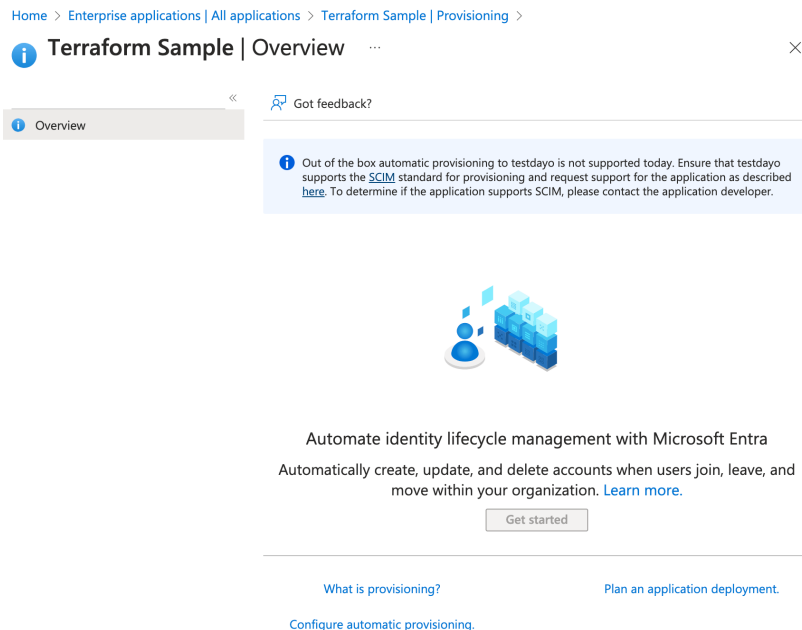
```
resource "azuread_named_location" "ip_google_dns" {
  display_name = "Google DNS"
  ip {
    ip_ranges = [
      "8.8.8.8/32",
    ]
    trusted = true
  }
}
```

運用してみて分かってきたこと、工夫したこととか

# プロビジョニング設定ができなくなる問題

単純にサービスプリンシパルを作成してしまうと、プロビジョニング設定ができなくなってしまう

サービスプリンシパルにプロビジョニング構成のテンプレートを紐付けるMicrosoft Graph APIが存在しないのが諸悪の根源



The screenshot shows the Microsoft Entra admin center interface for the 'Terraform Sample' application. The breadcrumb trail is 'Home > Enterprise applications | All applications > Terraform Sample | Provisioning > Terraform Sample | Overview'. A navigation bar includes 'Overview' and a 'Got feedback?' link. A blue information banner displays the error: 'Out of the box automatic provisioning to testdayo is not supported today. Ensure that testdayo supports the SCIM standard for provisioning and request support for the application as described here. To determine if the application supports SCIM, please contact the application developer.' Below the banner is a blue icon representing identity management. The main content area features the heading 'Automate identity lifecycle management with Microsoft Entra' and the text 'Automatically create, update, and delete accounts when users join, leave, and move within your organization. [Learn more.](#)' with a 'Get started' button. At the bottom, there are links for 'What is provisioning?', 'Plan an application deployment.', and 'Configure automatic provisioning.'

# プロビジョニング設定ができなくなる問題

## 解決方法

ギャラリーテンプレートを使うようにする  
パターン

```
data "azuread_application_template" "marketo" {
  display_name = "Marketo"
}

resource "azuread_application" "marketo" {
  display_name = "Marketo"
  template_id  = data.azuread_application_template.marketo
}

resource "azuread_service_principal" "marketo" {
  use_existing = true
  client_id    = azuread_application.marketo.client_id
}
```

Terraform外で作成してインポートするパ  
ターン

```
import {
  id = "ここにアプリケーションのObject ID"
  to = azuread_application.marketo
}

import {
  id = "ここにサービスプリンシパルのObject ID"
  to = azuread_service_principal.marketo
}

resource "azuread_application" "marketo" {
  display_name = "Marketo"
}

resource "azuread_service_principal" "marketo" {
  client_id = azuread_application.marketo.client_id
}
```

# 特定の権限でないとグループの操作ができない問題

APIの呼び出し権限には`Delegated`と`Application`が存在するが、一部属性のみ`Delegated`のみがサポート[1]

つまり、CI/CDで実行できなくなってしまう

- `external\_senders\_allowed`
- `auto\_subscribe\_new\_members`
- `hide\_from\_address\_lists`
- `hide\_from\_outlook\_clients`

```
provider "azuread" {  
  tenant_id      = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxx"  
  client_id      = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxx"  
  client_secret  = "xxxx"  
}
```

```
data "azuread_group" "bizdev_division" {  
  display_name = "BizDev Division"  
}
```

```
resource "azuread_service_principal" "marketo" {  
  owners = data.azuread_group.bizdev_division.members  
}
```

1. Microsoft側の既知の問題として認識されていたが、いまはそのIssueも参照できない状態 [🔗](#)



# 特定の権限でないとグループの操作ができない問題

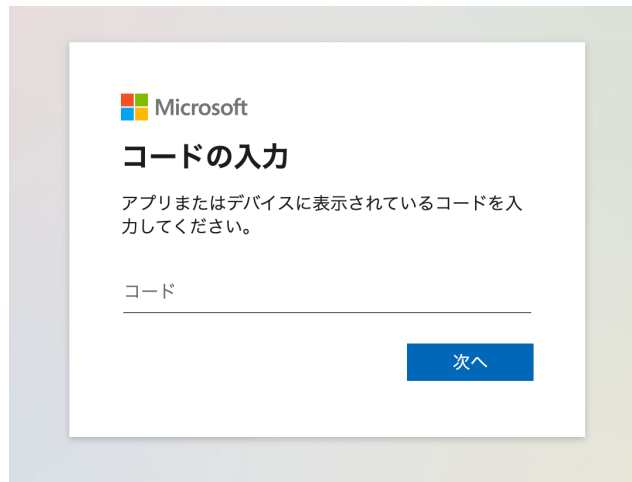
Azure CLIのデバイス認証で代用[!]


-----

Terraform Provider for Azure Active DirectoryはAzure CLIの認証情報を使える

```
# CI/CDでの処理
$ az login --allow-no-subscription --use-device-code
To sign in, use a web browser to open the page https://mic

$ terraform init && terraform plan
```



1. 毎回CI/CDが実行されるたびに認証をしないとイケない、というデメリットがある 

# 特定の権限でないとグループの操作ができない問題

複数の認証を使用することで、影響を最小限にできる

```
provider "azuread" {
  tenant_id      = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx"
  client_id      = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx"
  client_secret  = "xxxx"
  use_cli        = false
}

provider "azuread" {
  alias          = "device_auth"
  tenant_id     = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx"
}
```

```
data "azuread_group" "bizdev_division" {
  provider      = azuread.device_auth
  display_name  = "BizDev Division"
}

resource "azuread_service_principal" "marketo" {
  owners = data.azuread_group.bizdev_division.members
}
```

# Terraformで作成したリソースがわかるようにする

Terraformで作成したリソースとそうでないリソースを、Azure Portalなどで区別できるようにすると管理的に少しだけはかどる

```
locals {
  managed_by_terraform = "Managed by Terraform"
}

resource "azuread_application" "google_workspace" {
  display_name = "Google Workspace"
  notes       = local.managed_by_terraform
}

resource "azuread_group" "bizdev_division" {
  display_name = "BizDev Division"
  description  = "BizDev Division ${local.managed_by_terraform}"
}
```

本当はAWSみたいに、各リソースにタグ的なものが存在してほしい

# SAMLの証明書をアップロードする必要がある場合

Blue/Greenデプロイ的な感じで証明書を切り替えるためのモジュールを用意する

```
variable "service_principal_id" {  
  type      = string  
  description = "証明書を紐付けるためのサービスプリンシパルのオブシ  
}
```

```
variable "certificate_end_dates" {  
  type = object({  
    blue = optional(string)  
    green = optional(string)  
  })  
  description = <<EOS  
    blueまたはgreenには証明書の有効期限の日付（RFC3339のUTCフォー
```

azuread\_service\_principal\_token\_signing\_certificateリソ  
サービスプロバイダ側にアップロードなどで証明書を登録する必要がある  
この挙動の場合は、作成された証明書をサービスプロバイダ側に登録する

そのため、ブルーグリーンデプロイメントの要領で証明書を更新できるよ

```
resource "azuread_service_principal_token_signing_certific  
  count = var.certificate_end_dates.blue == null ? 0 : 1  
  
  service_principal_id = var.service_principal_id  
  end_date             = var.certificate_end_dates.blue  
}
```

```
resource "azuread_service_principal_token_signing_certific  
  count = var.certificate_end_dates.green == null ? 0 : 1  
  
  service_principal_id = var.service_principal_id  
  end_date             = var.certificate_end_dates.green  
}
```

# SAMLの証明書をアップロードする必要がある場合

1. Blueの証明書を用意する
2. Blueの証明書をSPにアップロードする
3. Blueの証明書を有効化する

```
module "marketo_certificate" {  
  source           = "./modules/service-principal-token-signing-certificate"  
  service_principal_id = azuread_service_principal.marketo.id  
  certificate_end_dates = {  
    blue = "2024-03-31T14:59:59Z"  
  }  
}
```

1. Greenの証明書を用意する
2. Greenの証明書をSPにアップロードする
3. Greenの証明書を有効化する

```
module "marketo_certificate" {  
  source           = "./modules/service-principal-token-signing-certificate"  
  service_principal_id = azuread_service_principal.marketo.id  
  certificate_end_dates = {  
    blue  = "2024-03-31T14:59:59Z"  
    green = "2026-03-31T14:59:59Z"  
  }  
}
```

1. Blueの証明書を削除する

```
module "marketo_certificate" {  
  source           = "./modules/service-principal-token-signing-certificate"  
  service_principal_id = azuread_service_principal.marketo.id  
  certificate_end_dates = {  
    green = "2026-03-31T14:59:59Z"  
  }  
}
```

# グループのメンバーアサインを変更する場合は追加と削除を分ける

サービスプリンシパルにグループをアサインして、かつプロビジョニングをしていると、長い処理の場合、一時的にメンバーがいなくなってしまう

Terraformに限った話ではないが、運悪くプロビジョニングが先行して走ってしまい、ユーザーが無効化するなんてことになる

# グループの作成でコケるとめんどくさい


Microsoft Entra IDのグループ作成に関連して作成されるリソース（例えばSharePointのサイト）の名前に余計な接頭辞がついてしまう

グループの作成時に、Exchange Onlineに紐づく値や、メンバー情報なども一貫したトランザクションがない状態でリソースを操作しているため[!]

残念ながらどうしようもないので、そういうことが起こるという事実を覚えておいて、コケてしまったら、以下のように手動でなんとかするしかない

- グループを一度削除する
- 関連して作成されたリソースの名前を変更しに行く

---

1. [terraform-provider-azuread/internal/services/groups/group\\_resource.go](https://github.com/terraform-provider-azuread/internal/services/groups/group_resource.go) at main · hashicorp/terraform-provider-azuread 

# 賛否両論な運用例たち

- 動的グループ使うと個別アサインできないので、Terraformで頑張っちゃう
- コンポーネントごとにstateファイルを分割する
- PIMなど対応してないリソースを頑張っ  
て対応する

```
resource "azuread_group" "business_unit" {  
  display_name = "Business Unit"  
  members      = concat([  
    azuread_user.kenchan0130.object_id,  
  ], data.azuread_group.bizdev_division.members)  
}
```

```
.  
├─ application  
├─ group  
└─ pim
```

```
resource "terraform_data" "pim_global_admin" {  
  # 実行するコマンド  
  provisioner "local-exec" {  
    command = "curl ....."  
  }  
}
```



# 終わりに

- 運用すると色々考慮しないといけないことがいっぱいある
- だれが何をしたのか、その意図は何だっけをどれだけ残したいかとの天秤
- Microsoft Entra IDをTerraformで管理する人口が増える一助になれば嬉しい

# 添付: 構成管理のTerraform以外の方法

- [Microsoft365DSC](#)
- [!\[\]\(467d80e979964f7f8c752fb22248b5b7\_img.jpg\) Pulumi](#)

# 添付: Intuneサポートを追加しようとしたが、アプリケーションのファイルアップロードが実現できなくて困っている

ファイルの暗号化をしてアップロードしないといけませんが、暗号化の仕様がよくわからない

- [Create mobileAppContentFile - Microsoft Graph](#)
- [commit action - Microsoft Graph](#)

サポートに問い合わせもしたが、[!\[\]\(4729e517bc6a7cd81c8025b9646574fb\_img.jpg\) microsoftgraph/powershell-intune-samples](#)見てねしか答えられないわ！（意訳）って言われてしまった