Deploy policies without enforcing them (Level 300)

By:
Sarfaraz Momin & Sarang Pardeshi
(Cloud Engineer)

Strategy

Policies have a property enforcementMode. The enforcementMode property provides customers the ability to test the outcome of a policy on existing resources without initiating the policy effect or triggering entries in the Azure Activity log.

This scenario is commonly referred to as "What If" and aligns to safe deployment practices. enforcementMode is different from the Disabled effect, as that effect prevents resource evaluation from happening at all.

Setting the enforcementMode property to false can be useful in browncase scenarios, so that existing workloads can be changed before enforcing the policies.

- **terraform.tf**
- **variables.tf**
- **main.tf**
- **archetype_config_overrides.tf**

--CODE--

terraform.tf

```
# Configure Terraform to set the required AzureRM provider
# version and features{} block.

terraform {
  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = ">= 3.19.0"
    }
  }
}

provider "azurerm" {
  features {}
}
```

variables.tf

```
# Use variables to customize the deployment

variable "root_id" {
  type    = string
  default = "myorg"
}

variable "root_name" {
  type    = string
  default = "My Organization"
}
```

main.tf

```
# You can use the azurerm_client_config data resource to dynamically
# extract connection settings from the provider configuration.

data "azurerm_client_config" "core" {}

# Call the caf-enterprise-scale module directly from the Terraform Registry
# pinning to the latest version

module "enterprise_scale" {
  source  = "Azure/caf-enterprise-scale/azurerm"
  version = "4.0.1" # change this to your desired version,
https://www.terraform.io/language/expressions/version-constraints, should be at least 3.4.0

  default_location = "eastus"

  providers = {
    azurerm              = azurerm
```

```
    azurerm.connectivity = azurerm
    azurerm.management   = azurerm
  }

  root_parent_id = data.azurerm_client_config.core.tenant_id
  root_id        = "myorg"
  root_name      = "My Organization"

  deploy_corp_landing_zones   = true
  deploy_online_landing_zones = true
  deploy_identity_resources   = true

  archetype_config_overrides = local.archetype_config_overrides
}
```

Please edit `version = "<VERSION>"` **&** `default_location = "YOUR_LOCATION"`

archetype_config_overrides.tf

```
locals {
    myorg-landing-zones = {
      enforcement_mode = {
        Deny-IP-Forwarding       = false
        Deny-RDP-From-Internet   = false
        Deny-Storage-http        = false
        Deny-Subnet-Without-Nsg  = false
        Deploy-AKS-Policy        = false
        Deploy-SQL-DB-Auditing   = false
        Deploy-SQL-Threat        = false
        Deploy-VM-Backup         = false
        Deny-Priv-Escalation-AKS = false
        Deny-Priv-Containers-AKS = false
        Enable-DDoS-VNET         = false
        Enforce-AKS-HTTPS        = false
        Enforce-TLS-SSL          = false
      }
    }
  }
}
```

Number of Policies in This Deployment.

Count: 382

Number of Policy Assignments in This Deployment.

Count: 13

Number of Resources in This Deployment.

Count: 0

Tree Structure:

```
rootUST# tree test04
test04
├── archetype_config_overrides.tf
├── main.tf
├── terrform.tf
└── variables.tf

0 directories, 4 files
rootUST#
```

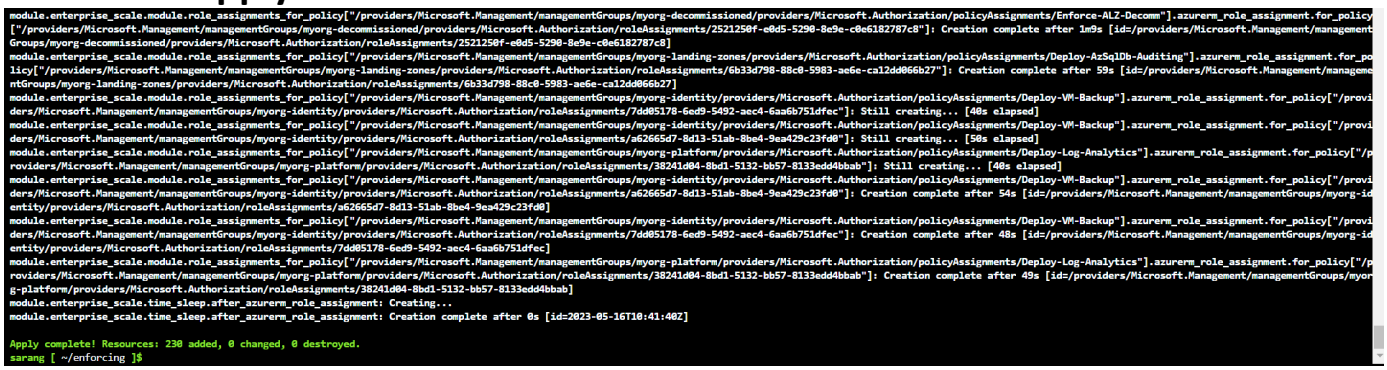
Terraform Init:

```
- Finding hashicorp/azurerm versions matching ">= 3.19.0, >= 3.54.0"...
- Installing hashicorp/time v0.9.1...
- Installed hashicorp/time v0.9.1 (signed by HashiCorp)
- Installing hashicorp/random v3.5.1...
- Installed hashicorp/random v3.5.1 (signed by HashiCorp)
- Installing azure/azapi v1.6.0...
- Installed azure/azapi v1.6.0 (signed by a HashiCorp partner, key ID 6F0B91BDE98478CF)
- Installing hashicorp/azurerm v3.56.0...
- Installed hashicorp/azurerm v3.56.0 (signed by HashiCorp)

Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
sarfaraz [ ~/test04 ]$
```

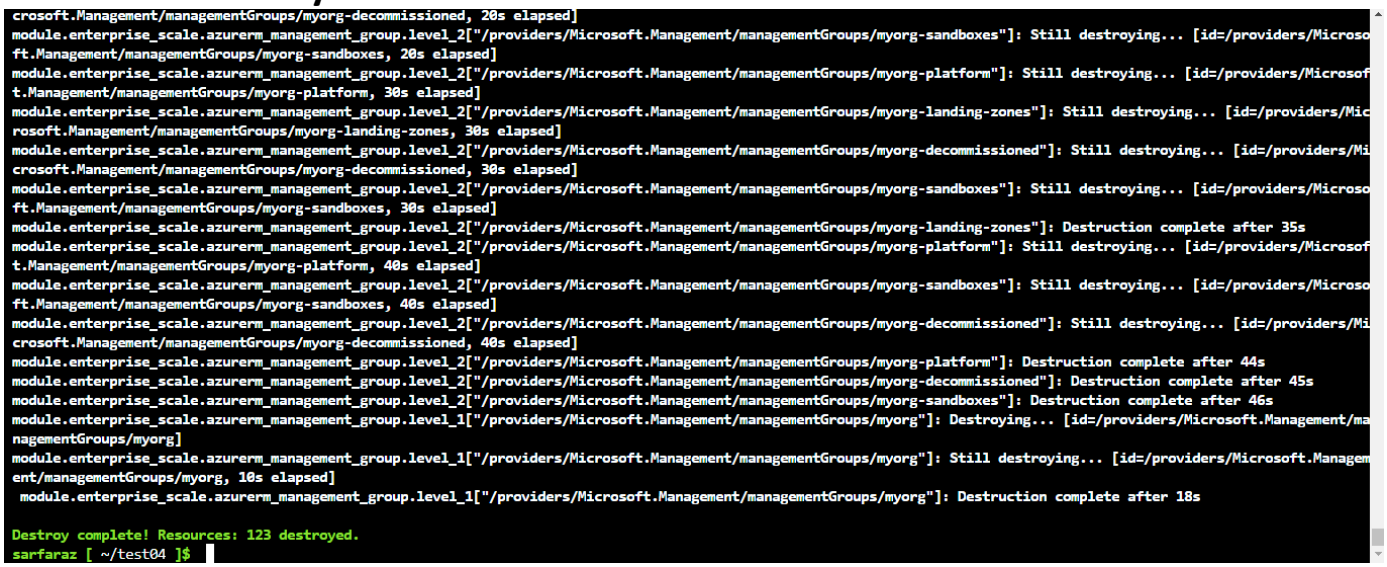
Terraform Plan:

```
      + name                           = "55ffe1be-e389-5d46-9488-8d6915a8b60e"
      + principal_id                   = (known after apply)
      + principal_type                 = (known after apply)
      + role_definition_id             = "/providers/Microsoft.Authorization/roleDefinitions/92aaf0da-9dab-42b6-94a3-d43ce8d16293"
      + role_definition_name           = (known after apply)
      + scope                          = "/providers/Microsoft.Management/managementGroups/myorg"
      + skip_service_principal_aad_check = (known after apply)
    }

  # module.enterprise_scale.module.role_assignments_for_policy["/providers/Microsoft.Management/managementGroups/myorg/providers/Microsoft.Authorization/policyAssignments/Enforce-ACSB"].azurerm_role_assignment.for_policy["/providers/Microsoft.Management/managementGroups/myorg/providers/Microsoft.Authorization/roleAssignments/20b87dbc-9b70-5379-ad61-97a3ccecc927"] will be created
  + resource "azurerm_role_assignment" "for_policy" {
      + id                           = (known after apply)
      + name                         = "20b87dbc-9b70-5379-ad61-97a3ccecc927"
      + principal_id                 = (known after apply)
      + principal_type               = (known after apply)
      + role_definition_id           = "/providers/Microsoft.Authorization/roleDefinitions/b24988ac-6180-42a0-ab88-20f7382dd24c"
      + role_definition_name         = (known after apply)
      + scope                        = "/providers/Microsoft.Management/managementGroups/myorg"
      + skip_service_principal_aad_check = (known after apply)
    }

Plan: 230 to add, 0 to change, 0 to destroy.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
sarfaraz [ ~/test04 ]$
```

Terraform Apply:

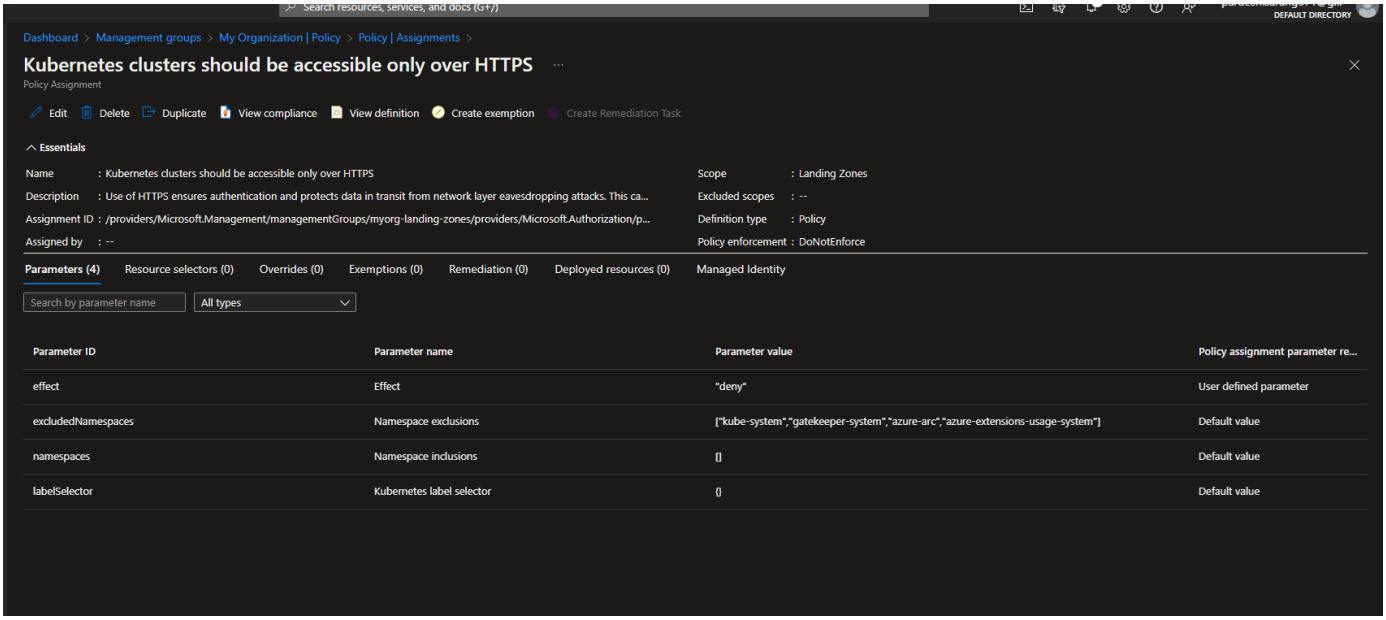


Terraform Destroy:



--SCREEN'S--

Kubernetes cluster should be accessible only over HTTPS.



--END--