

[Open in app](#)

Guillermo Musumeci

[Follow](#)

497 Followers

[About](#)Picture by [moren hsu](#) at [Unsplash](#)

How to Manage Azure Key Vault with Terraform

[Guillermo Musumeci](#) Jan 14, 2020 · 2 min read ★

The purpose of **Azure Key Vault** is to store cryptographic keys and other secrets used by cloud apps and services in a **HSM** (Hardware security module). A **HSM** is a physical

[Open in app](#)

In this example, we will create a **Terraform** module to manage an **Azure Key Vault**.

A Terraform module is used to package or encapsulate multiple resources together. Modules simplify the re-utilization of code and can be called from other modules or from the root module.

If you don't want to use modules, just copy the module files to the root folder of your **Terraform** deployment.

Azure Key Vault Module:

Inside the **keyvault** folder, create the **variables.tf** file to store variables used by the module:

```
1  #####
2  # Azure Resource Group variables #
3  #####
4
5  variable "resource_group_name" {
6      type          = string
7      description = "The name of an existing Resource Group"
8  }
9
10 variable "location" {
11     type          = string
12     description = "Define the region the Azure Key Vault should be created, you should use the R
13 }
14
15 #####
16 # Azure Key Vault variables #
17 #####
18
19 variable "name" {
20     type          = string
21     description = "The name of the Azure Key Vault"
22 }
23
24 variable "sku_name" {
25     type          = string
26     description = "Select Standard or Premium SKU"
27     default      = "standard"
28 }
```

[Open in app](#)

```
32     description = "Allow Azure Virtual Machines to retrieve certificates stored as secrets from
33     default     = "true"
34 }
35
36 variable "enabled_for_disk_encryption" {
37     type      = string
38     description = "Allow Azure Disk Encryption to retrieve secrets from the Azure Key Vault and
39     default   = "true"
40 }
41
42 variable "enabled_for_template_deployment" {
43     type      = string
44     description = "Allow Azure Resource Manager to retrieve secrets from the Azure Key Vault"
45     default   = "true"
46 }
47
48 variable "kv-key-permissions-full" {
49     type      = list(string)
50     description = "List of full key permissions, must be one or more from the following: backup,
51     default    = [ "backup", "create", "decrypt", "delete", "encrypt", "get", "import", "list",
52                  "recover", "restore", "sign", "unwrapKey", "update", "verify", "wrapKey" ]
53 }
54
55 variable "kv-secret-permissions-full" {
56     type      = list(string)
57     description = "List of full secret permissions, must be one or more from the following: back
58     default    = [ "backup", "delete", "get", "list", "purge", "recover", "restore", "set" ]
59 }
60
61 variable "kv-certificate-permissions-full" {
62     type      = list(string)
63     description = "List of full certificate permissions, must be one or more from the following:
64     default    = [ "create", "delete", "deleteissuers", "get", "getissuers", "import", "list",
65                  "managecontacts", "manageissuers", "purge", "recover", "setissuers", "update
66 }
67
68 variable "kv-storage-permissions-full" {
69     type      = list(string)
70     description = "List of full storage permissions, must be one or more from the following: bac
71     default    = [ "backup", "delete", "deletesas", "get", "getsas", "list", "listsas",
72                  "purge", "recover", "regeneratekey", "restore", "set", "setsas", "update" ]
73 }
74
75 variable "kv-key-permissions-read" {
76     type      = list(string)
```

[Open in app](#)

```
80
81 variable "kv-secret-permissions-read" {
82     type          = list(string)
83     description = "List of full secret permissions, must be one or more from the following: back
84     default       = [ "get", "list" ]
85 }
86
87 variable "kv-certificate-permissions-read" {
88     type          = list(string)
89     description = "List of full certificate permissions, must be one or more from the following:
90     default       = [ "get", "getissuers", "list", "listissuers" ]
91 }
92
93 variable "kv-storage-permissions-read" {
94     type          = list(string)
95     description = "List of read storage permissions, must be one or more from the following: bac
96     default       = [ "get", "getsas", "list", "listsas" ]
97 }
98
99 variable "tags" {
100     description = "A mapping of tags to assign to the resource"
101     type        = map(string)
102     default     = {}
103 }
104
105 variable "policies" {
106     type = map(object({
107         tenant_id          = string
108         object_id          = string
109         key_permissions    = list(string)
110         secret_permissions = list(string)
111         certificate_permissions = list(string)
112         storage_permissions = list(string)
113     }))
114     description = "Define a Azure Key Vault access policy"
115     default = {}
116 }
117
118 variable "secrets" {
119     type = map(object({
120         value = string
121     }))
122     description = "Define Azure Key Vault secrets"
123     default = {}
124 }
```

[Open in app](#)

Then, create the main file to create the Azure Key Vault and policies, inside the key-vault folder:

```
1  data "azurerm_client_config" "current" {}
2
3  # Create the Azure Key Vault
4  resource "azurerm_key_vault" "key-vault" {
5      name                = var.name
6      location            = var.location
7      resource_group_name = var.resource_group_name
8
9      enabled_for_deployment      = var.enabled_for_deployment
10     enabled_for_disk_encryption = var.enabled_for_disk_encryption
11     enabled_for_template_deployment = var.enabled_for_template_deployment
12
13     tenant_id = data.azurerm_client_config.current.tenant_id
14     sku_name  = var.sku_name
15     tags      = var.tags
16
17     network_acls {
18         default_action = "Allow"
19         bypass         = "AzureServices"
20     }
21 }
22
23 # Create a Default Azure Key Vault access policy with Admin permissions
24 # This policy must be kept for a proper run of the "destroy" process
25 resource "azurerm_key_vault_access_policy" "default_policy" {
26     key_vault_id = azurerm_key_vault.key-vault.id
27     tenant_id    = data.azurerm_client_config.current.tenant_id
28     object_id    = data.azurerm_client_config.current.object_id
29
30     lifecycle {
31         create_before_destroy = true
32     }
33
34     key_permissions = var.kv-key-permissions-full
35     secret_permissions = var.kv-secret-permissions-full
36     certificate_permissions = var.kv-certificate-permissions-full
37     storage_permissions = var.kv-storage-permissions-full
38 }
39
40 # Create an Azure Key Vault access policy
41 resource "azurerm_key_vault_access_policy" "policy" {
```

[Open in app](#)

```
45     object_id           = lookup(each.value, "object_id")
46     key_permissions     = lookup(each.value, "key_permissions")
47     secret_permissions  = lookup(each.value, "secret_permissions")
48     certificate_permissions = lookup(each.value, "certificate_permissions")
49     storage_permissions  = lookup(each.value, "storage_permissions")
50 }
51
52 # Generate a random password
53 resource "random_password" "password" {
54     for_each      = var.secrets
55     length        = 20
56     min_upper     = 2
57     min_lower     = 2
58     min_numeric   = 2
59     min_special   = 2
60
61     keepers = {
62         name = each.key
63     }
64 }
65
66 # Create an Azure Key Vault secrets
67 resource "azurerm_key_vault_secret" "secret" {
68     for_each      = var.secrets
69     key_vault_id = azurerm_key_vault.key-vault.id
70     name          = each.key
71     value         = lookup(each.value, "value") != "" ? lookup(each.value, "value") : random_password.value
72     tags          = var.tags
73     depends_on = [
74         azurerm_key_vault.key-vault,
75         azurerm_key_vault_access_policy.default_policy,
76     ]
77 }
```

Finally, we create the **output.tf** file in the same folder used to return the values of the **Terraform** module.

```
1  output "key-vault-id" {
2      description = "Key Vault ID"
3      value       = azurerm_key_vault.key-vault.id
4  }
5
```

[Open in app](#)

```
9   }
10
11   output "key-vault-secrets" {
12     value = values(azurerm_key_vault_secret.secret).*value
13   }
```

keyvault-output.tf hosted with ❤ by GitHub

[view raw](#)

How to use the Key Vault Module

Create the **variables.tf** file to store variables used by the module in the root folder. I use variables in a separate **variables.tf** file for readability, however, you can use variables directly in the **main.tf** code.

```
1  #####
2  # Key Vault variables #
3  #####
4
5  variable "kv-full-object-id" {
6    type          = string
7    description = "The object ID of a user, service principal or security group in the Azure Acti
8    default       = ""
9  }
10
11 variable "kv-read-object-id" {
12   type          = string
13   description = "The object ID of a user, service principal or security group in the Azure Acti
14   default       = ""
15 }
16
17 variable "kv-vm-deployment" {
18   type          = string
19   description = "Allow Azure Virtual Machines to retrieve certificates stored as secrets from t
20   default       = "true"
21 }
22
23 variable "kv-disk-encryption" {
24   type          = string
25   description = "Allow Azure Disk Encryption to retrieve secrets from the Azure Key Vault and u
26   default       = "true"
27 }
28
29 variable "kv-template-deployment" {
30   type          = string
```

[Open in app](#)

```
34
35 variable "kv-key-permissions-full" {
36     type          = list(string)
37     description = "List of full key permissions, must be one or more from the following: backup,
38     default      = [ "backup", "create", "decrypt", "delete", "encrypt", "get", "import", "list",
39                     "recover", "restore", "sign", "unwrapKey","update", "verify", "wrapKey" ]
40 }
41
42 variable "kv-secret-permissions-full" {
43     type          = list(string)
44     description = "List of full secret permissions, must be one or more from the following: backu
45     default      = [ "backup", "delete", "get", "list", "purge", "recover", "restore", "set" ]
46 }
47
48 variable "kv-certificate-permissions-full" {
49     type          = list(string)
50     description = "List of full certificate permissions, must be one or more from the following:
51     default      = [ "create", "delete", "deleteissuers", "get", "getissuers", "import", "list", '
52                     "managecontacts", "manageissuers", "purge", "recover", "setissuers", "update'
53 }
54
55 variable "kv-storage-permissions-full" {
56     type          = list(string)
57     description = "List of full storage permissions, must be one or more from the following: back
58     default      = [ "backup", "delete", "deletesas", "get", "getsas", "list", "listsas",
59                     "purge", "recover", "regeneratekey", "restore", "set", "setsas", "update" ]
60 }
61
62 variable "kv-key-permissions-read" {
63     type          = list(string)
64     description = "List of read key permissions, must be one or more from the following: backup,
65     default      = [ "get", "list" ]
66 }
67
68 variable "kv-secret-permissions-read" {
69     type          = list(string)
70     description = "List of full secret permissions, must be one or more from the following: backu
71     default      = [ "get", "list" ]
72 }
73
74 variable "kv-certificate-permissions-read" {
75     type          = list(string)
76     description = "List of full certificate permissions, must be one or more from the following:
77     default      = [ "get", "getissuers", "list", "listissuers" ]
78 }
```


[Open in app](#)

```

81     type      = list(string)
82     description = "List of read storage permissions, must be one or more from the following: back
83     default     = [ "get", "getsas", "list", "listsas" ]
84 }
85
86 variable "kv-secrets" {
87     type = map(object({
88         value = string
89     }))
90     description = "Define Azure Key Vault secrets"
91     default     = {}
92 }

```

Here is the code to call the module in the **main.tf** file of the root folder. We will pass the Resource Group, location and other parameters to the module.

Also, we created two Key Vault policies, one full for administrators and one read for applications. The settings of these policies can be modified from the **variables.tf** file.

```

1  module "keyvault" {
2      source          = "./modules/keyvault"
3      name            = "${var.environment}-keyvault"
4      location        = azurerm_resource_group.security-rg.location
5      resource_group_name = azurerm_resource_group.security-rg.name
6
7      enabled_for_deployment      = var.kv-vm-deployment
8      enabled_for_disk_encryption = var.kv-disk-encryption
9      enabled_for_template_deployment = var.kv-template-deployment
10
11      tags = {
12          environment = "${var.environment}"
13      }
14
15      policies = {
16          full = {
17              tenant_id          = var.azure-tenant-id
18              object_id          = var.kv-full-object-id
19              key_permissions     = var.kv-key-permissions-full
20              secret_permissions = var.kv-secret-permissions-full
21              certificate_permissions = var.kv-certificate-permissions-full
22              storage_permissions = var.kv-storage-permissions-full
23          }
24          read = {

```

[Open in app](#)

```

27     key_permissions      = var.kv-key-permissions-read
28     secret_permissions   = var.kv-secret-permissions-read
29     certificate_permissions = var.kv-certificate-permissions-read
30     storage_permissions   = var.kv-storage-permissions-read
31   }
32 }
33
34 secrets = var.kv-secrets
35 }

```

keyvault-root-main.tf hosted with ❤ by GitHub

[view raw](#)

Finally, we define the content variables in the **terraform.tfvars** file. We will pass the **object ID** of a user, service principal or security group for FULL and READ access using **kv-full-object-id** and **kv-read-object-id** variables and the **secrets** using a map object.

```

1 kv-full-object-id = ""
2 kv-read-object-id = ""
3 kv-secrets = {
4   sqldb = {
5     value = "" # setting to "" will auto-generate the password
6   }
7   webadmin = {
8     value = "hLDmexfL8@m46Suevb!oao"
9   }
10 }

```

keyvault-terraform.tfvars.tf hosted with ❤ by GitHub

[view raw](#)

(Optionally) if we want to see the output of the module, we can add an **output.tf** file to the root folder with the following content:

```

1 output "key-vault-id" {
2   description = "Key Vault ID"
3   value       = module.keyvault.key-vault-id
4 }
5
6 output "key-vault-url" {
7   description = "Key Vault URI"
8   value       = module.keyvault.key-vault-url
9 }
10
11 output "key-vault-secrets" {
12   description = "Key Vault Secrets"

```

[Open in app](#)

15

keyvault-root-output.tf hosted with ❤ by GitHub

[view raw](#)

The full code of this example is available at my **GitHub** repository

<https://github.com/guillermo-musumeci/terraform-azure-key-vault-module/>

And that's all folks. Thank you for reading!

[Azure](#)[Keyvault](#)[Infrastructure As Code](#)[Terraform](#)[About](#) [Help](#) [Legal](#)

Get the Medium app

