» vcd_lb_service_monitor

Provides a vCloud Director Edge Gateway Load Balancer Service Monitor data source. A service monitor defines health check parameters for a particular type of network traffic. It can be associated with a pool. Pool members are monitored according to the service monitor parameters. See example usage of this data source in server pool resource page.

Note: See additional support notes in service monitor resource page.

Supported in provider *v2.4+*

» Example Usage

```
data "vcd_lb_service_monitor" "my-monitor" {
  org                  = "my-org"
  vdc                  = "my-org-vdc"
  edge_gateway         = "my-edge-gw"

  name = "not-managed"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `edge_gateway` - (Required) The name of the edge gateway on which the service monitor is defined
- `name` - (Required) Service Monitor name for identifying the exact service monitor

» Attribute Reference

All the attributes defined in `vcd_lb_service_monitor` resource are available.

» vcd_lb_server_pool

Provides a vCloud Director Edge Gateway Load Balancer Server Pool data source. A Server Pool defines a group of backend servers (defined as pool members), manages load balancer distribution methods, and has a service monitor attached to it for health check parameters.

Note: See additional support notes in server pool resource page.

Supported in provider *v2.4+*

» Example Usage

```
data "vcd_lb_server_pool" "sp-ds" {
  org                = "my-org"
  vdc                = "my-org-vdc"
  edge_gateway       = "my-edge-gw"

  name = "not-managed"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `edge_gateway` - (Required) The name of the edge gateway on which the server pool is defined
- `name` - (Required) Server Pool name for identifying the exact server pool

» Attribute Reference

All the attributes defined in `vcd_lb_server_pool` resource are available.

» vcd_lb_app_profile

Provides a vCloud Director Edge Gateway Load Balancer Application Profile data source. An application profile defines the behavior of the load balancer for a particular type of network traffic. After configuring a profile, you associate it with a virtual server. The virtual server then processes traffic according to the values specified in the profile.

Note: See additional support notes in application profile resource page.

Supported in provider *v2.4+*

» Example Usage

```
data "vcd_lb_app_profile" "my-profile" {
  org                = "my-org"
  vdc                = "my-org-vdc"
  edge_gateway       = "my-edge-gw"

  name = "not-managed"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `edge_gateway` - (Required) The name of the edge gateway on which the service monitor is defined
- `name` - (Required) Application profile name for identifying the exact application profile

» Attribute Reference

All the attributes defined in `vcd_lb_app_profile` resource are available.

» vcd_lb_app_rule

Provides a vCloud Director Edge Gateway Load Balancer Application Rule data source. An application rule allows to directly manipulate and manage IP application traffic with load balancer.

Note: See additional support notes in application rule resource page.

Supported in provider *v2.4+*

» Example Usage

```
data "vcd_lb_app_rule" "my-rule" {
  org                 = "my-org"
  vdc                 = "my-org-vdc"
  edge_gateway        = "my-edge-gw"

  name = "not-managed"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `edge_gateway` - (Required) The name of the edge gateway on which the service monitor is defined
- `name` - (Required) Application rule name for identifying the exact application rule

» Attribute Reference

All the attributes defined in `vcd_lb_app_rule` resource are available.

» vcd_lb_virtual_server

Provides a vCloud Director edge gateway load balancer virtual server data source. Adds an edge gateway internal or uplink interface as a virtual server. A virtual server has a public IP address and services all incoming client requests.

Note: To make load balancing work one must ensure that load balancing is enabled on edge gateway (edge gateway must be advanced). This depends on NSX version to work properly. Please refer to VMware Product Interoperability Matrices to check supported vCloud director and NSX for vSphere configurations.

Note: The vCloud Director API for NSX supports a subset of the operations and objects defined in the NSX vSphere API Guide. The API supports NSX 6.2, 6.3, and 6.4.

Supported in provider *v2.4+*

» Example Usage

```
data "vcd_lb_virtual_server" "my-vs" {
  org                 = "my-org"
  vdc                 = "my-org-vdc"
  edge_gateway        = "my-edge-gw"

  name = "not-managed"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `edge_gateway` - (Required) The name of the edge gateway on which the virtual server is defined
- `name` - (Required) Name for identifying the exact virtual server

» Attribute Reference

All the attributes defined in `vcd_lb_virtual_server` resource are available.

» vcd_nsxv_dnat

Provides a vCloud Director DNAT data source for advanced edge gateways (NSX-V). This can be used to read existing rule by ID and use its attributes in other resources.

Note: This data source requires advanced edge gateway. For non-advanced edge gateways please use the `vcd_dnat` resource.

» Example Usage

```
data "vcd_nsxv_dnat" "my-rule" {
  org                 = "my-org"
  vdc                 = "my-org-vdc"
  edge_gateway        = "my-edge-gw"
  rule_id = "197864"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations.
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level.
- `edge_gateway` - (Required) The name of the edge gateway on which to apply the DNAT rule.
- `rule_id` - (Required) ID of DNAT rule as shown in the UI.

» Attribute Reference

All the attributes defined in `vcd_nsxv_dnat` resource are available.

» vcd_nsxv_snat

Provides a vCloud Director SNAT data source for advanced edge gateways (NSX-V). This can be used to read existing rule by ID and use its attributes in other resources.

Note: This data source requires advanced edge gateway. For non-advanced edge gateways please use the `vcd_snat` resource.

» Example Usage

```
data "vcd_nsxv_snat" "my-rule" {
  org                 = "my-org"
  vdc                 = "my-org-vdc"
  edge_gateway        = "my-edge-gw"

  rule_id = "197867"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations.
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level.
- `edge_gateway` - (Required) The name of the edge gateway on which to apply the SNAT rule.
- `rule_id` - (Required) ID of SNAT rule as shown in the UI.

» Attribute Reference

All the attributes defined in `vcd_nsxv_snat` resource are available.

» vcd_nsxv_firewall_rule

Provides a vCloud Director firewall rule data source for advanced edge gateways (NSX-V). This can be used to read existing rules by ID and use its attributes in other resources.

Note: This data source requires advanced edge gateway. For non-advanced edge gateways please use the `vcd_firewall_rules` resource.

» Example Usage

```
data "vcd_nsxv_firewall_rule" "my-rule" {
  org                = "my-org"
  vdc                = "my-org-vdc"
  edge_gateway       = "my-edge-gw"

  rule_id = "133048"  # real firewall rule ID, not the UI number
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations.

- vdc - (Optional) The name of VDC to use, optional if defined at provider level.
- edge_gateway - (Required) The name of the edge gateway on which to apply the DNAT rule.
- rule_id - (Required) ID of firewall rule (not UI number). See more information about firewall rule ID in vcd_nsxv_firewall_rule import section.

» Attribute Reference

All the attributes defined in vcd_nsxv_firewall_rule resource are available.

» vcd_org

Provides a vCloud Director Org data source. An organization can be used to manage catalogs, virtual data centers, and users.

Supported in provider *v2.5+*

» Example Usage

```
data "vcd_org" "my-org" {
  name   = "my-org"
}

resource "vcd_org" "my-org-clone" {
  name                 = "my-org-clone"
  full_name            = "${data.vcd_org.my-org.full_name}"
  can_publish_catalogs = "${data.vcd_org.my-org.can_publish_catalogs}"
  deployed_vm_quota    = "${data.vcd_org.my-org.deployed_vm_quota}"
  stored_vm_quota      = "${data.vcd_org.my-org.stored_vm_quota}"
  is_enabled           = "${data.vcd_org.my-org.is_enabled}"
  delete_force         = "true"
  delete_recursive     = "true"
}
```

» Argument Reference

The following arguments are supported:

- name - (Required) Org name

» Attribute Reference

- `full_name` - Org full name
- `is_enabled` - True if this organization is enabled (allows login and all other operations).
- `description` - Org description.
- `deployed_vm_quota` - Maximum number of virtual machines that can be deployed simultaneously by a member of this organization.
- `stored_vm_quota` - Maximum number of virtual machines in vApps or vApp templates that can be stored in an undeployed state by a member of this organization.
- `can_publish_catalogs` - True if this organization is allowed to share catalogs.
- `delay_after_power_on_seconds` - Specifies this organization's default for virtual machine boot delay after power on.

» vcd_org_vcd

Provides a vCloud Director Organization VDC data source. An Organization VDC can be used to reference a VCD and use its data within other resources or data sources.

Supported in provider *v2.5+*

» Example Usage

```
data "vcd_org_vdc" "my-org-vdc" {
  org     = "my-org"
  name    = "my-vdc"
}

output "provider_vdc" {
 value   = data.vcd_org_vdc.my-org-vdc.provider_vdc_name
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional, but required if not set at provider level) Org name
- `name` - (Required) Organization VDC name

» Attribute reference

All attributes defined in organization VDC resource are supported.

» vcd_catalog

Provides a vCloud Director Catalog data source. A Catalog can be used to manage catalog items and media items.

Supported in provider *v2.5+*

» Example Usage

```
data "vcd_catalog" "my-cat" {
  org  = "my-org"
  name = "my-cat"
}

resource "vcd_catalog_item" "myItem" {
  org     = "${data.vcd_catalog.my-cat.org}"
  catalog = "${data.vcd_catalog.my-cat.name}"

  name                 = "myItem"
  description          = "Belongs to ${data.vcd_catalog.my-cat.id}"
  ova_path             = "/path/to/test_vapp_template.ova"
  upload_piece_size    = 5
  show_upload_progress = "true"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional, but required if not set at provider level) Org name
- `name` - (Required) Catalog name

» Attribute Reference

- `description` - Catalog description.

» vcd_catalog_item

Provides a vCloud Director Catalog item data source. A Catalog item can be used to reference a catalog item and use its data within other resources or data sources.

Supported in provider *v2.5+*

» Example Usage

```
data "vcd_catalog_item" "my-first-item" {
  org     = "my-org"
  catalog = "my-cat"
  name    = "my-first-item"
}

resource "vcd_catalog_item" "my-second-item" {
  # Using the data source, two properties from another catalog items are
  # used in this resource.
  # You can read it as "use the org from catalog item `my-first-item`"
  # and "use the catalog from catalog item `my-first-item`"
  org     = "${data.vcd_catalog_item.my-first-item.org}"
  catalog = "${data.vcd_catalog_item.my-first-item.catalog}"

  name                 = "my-second-item"
  # The description uses the data source to create a dynamic text
  # The description will become "Belongs to my-cat"
  description          = "Belongs to ${data.vcd_catalog_item.my-first-item.catalog}"
  ova_path             = "/path/to/test_vapp_template.ova"
  upload_piece_size    = 5
  show_upload_progress = "true"
  metadata             = "${data.vcd_catalog_item.my-first-item.metadata}"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional, but required if not set at provider level) Org name
- `catalog` - (Required) Catalog name
- `name` - (Required) Catalog Item name

» Attribute Reference

- `description` - Catalog item description.
- `metadata` - Key value map of metadata.

» vcd_catalog_media

Provides a vCloud Director Catalog media data source. A Catalog media can be used to reference a catalog media and use its data within other resources or data sources.

Supported in provider *v2.5+*

» Example Usage

```
data "vcd_catalog_media" "existing-media" {
  org     = "my-org"
  catalog = "my-cat"
  name    = "my-media"
}

output "media_size" {
  value = data.vcd_catalog_media.existing-media.size
}

output "type_is_iso" {
  value = data.vcd_catalog_media.existing-media.is_iso
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level
- `catalog` - (Required) The name of the catalog where media file is
- `name` - (Required) Media name in catalog

» Attribute reference

All attributes defined in catalog_media are supported.

» vcd_independent_disk

Provides a vCloud Director Independent disk data source. A independent disk data source can be used to reference an independent disk and use its data within other resources or data sources.

Supported in provider *v2.5+*

» Example Usage

```
data "vcd_independent_disk" "existing-disk" {
  org     = "my-org"
  vdc     = "my-vdc"
  id      = "urn:vcloud:disk:1bbc273d-7701-4f06-97be-428b46b0805e"
  name    = "my-disk"
}
output "disk-iops" {
  value = data.vcd_independent_disk.existing-disk.iops
}
output "type_is_attached" {
  value = data.vcd_independent_disk.existing-disk.is_attached
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `id` - (Optional) Disk id or name is required. If both provided - Id is used. Id can be found by using import function Listing independent disk IDs
- `name` - (Optional) Disk name. **Warning** please use `id` as there is possibility to have more than one independent disk with same name. As result data source will fail.

» Attribute reference

All attributes defined in independent disk are supported.

» external_network

Provides a vCloud Director external network data source. This can be used to reference external networks and their properties.

Supported in provider *v2.5+*

» Example Usage

```
data "vcd_external_network" "tf-external-network" {
  name = "my-extnet"
}

resource "vcd_dnat" "tf-nat-rule" {
  org             = "tf-org"
  vdc             = "tf-vdc"
  # References the external network name from the data source
  network_name    = "${data.vcd_external_network.tf-external-network.name}"
  network_type    = "ext"
  edge_gateway    = "tf-gw"
  # References the first IP scope block. From that we extract the first static IP pool to re
  external_ip     = "${data.vcd_external_network.extnet-datacloud.ip_scope[0].static_ip_pool
  port            = 7777
  protocol        = "tcp"
  internal_ip     = "10.10.102.60"
  translated_port = 77
  description     = "test run"
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) external network name

» Attribute Reference

- `description` - Network friendly description
- `ip_scope` - A list of IP scopes for the network. See IP Scope for details.
- `vsphere_network` - A list of DV_PORTGROUP or NETWORK objects names that back this network. Each referenced DV_PORTGROUP or NETWORK must exist on a vCenter server registered with the system. See vSphere Network for details.

- `retain_net_info_across_deployments` - Specifies whether the network resources such as IP/MAC of router will be retained across deployments.

» vcd_edgegateway

Provides a vCloud Director edge gateway data source, directly connected to one or more external networks. This can be used to reference edge gateways for Org VDC networks to connect.

Supported in provider *v2.5+*

» Example Usage

```
data "vcd_edgegateway" "mygw" {
  name = "mygw"
  org  = "myorg"
  vdc  = "myvdc"
}

output "external_network" {
  value = data.vcd_edgegateway.mygw.default_gateway_network
}

# Get the name of the default gateway from the data source
# and use it to establish a second data source
data "vcd_external_network" "external_network1" {
  name = "${data.vcd_edgegateway.mygw.default_gateway_network}"
}

# From the second data source we extract the basic networking info
output "gateway" {
  value = data.vcd_external_network.external_network1.ip_scope.0.gateway
}
output "netmask" {
  value = data.vcd_external_network.external_network1.ip_scope.0.netmask
}
output "DNS" {
  value = data.vcd_external_network.external_network1.ip_scope.0.dns1
}
output "external_ip" {
  value = data.vcd_external_network.external_network1.ip_scope.0.static_ip_pool.0.start_addr
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) A unique name for the edge gateway.
- `org` - (Optional) The name of organization to which the VDC belongs. Optional if defined at provider level.
- `vdc` - (Optional) The name of VDC that owns the edge gateway. Optional if defined at provider level.

» Attribute Reference

All attributes defined in edge gateway resource are supported.

» vcd_network_routed

Provides a vCloud Director Org VDC routed Network data source. This can be used to reference internal networks for vApps to connect.

Supported in provider *v2.5+*

» Example Usage

```
data "vcd_network_routed" "net" {
  org  = "my-org" # Optional
  vdc  = "my-vdc" # Optional
  name = "my-net"
}

output "edge_gateway" {
  value = data.vcd_network_routed.net.edge_gateway
}

output "gateway" {
  value = data.vcd_network_routed.net.gateway
}
output "dhcp_start_address" {
  value = tolist(data.vcd_network_routed.net.dhcp_pool)[0].start_address
}

output "dhcp_end_address" {
  value = tolist(data.vcd_network_routed.net.dhcp_pool)[0].end_address
}
```

```
output "static_ip_start_address" {
  value = tolist(data.vcd_network_routed.net.static_ip_pool)[0].start_address
}

output "static_ip_end_address" {
  value = tolist(data.vcd_network_routed.net.static_ip_pool)[0].end_address
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `name` - (Required) A unique name for the network

» Attribute reference

All attributes defined in routed network resource are supported.

» vcd_network_isolated

Provides a vCloud Director Org VDC isolated Network data source. This can be used to reference internal networks for vApps to connect. This network is not attached to external networks or routers.

Supported in provider *v2.5+*

» Example Usage

```
data "vcd_network_isolated" "net" {
  org = "my-org" # Optional
  vdc = "my-vdc" # Optional

  name    = "my-net"
}

output "gateway" {
  value = data.vcd_network_isolated.net.gateway
}

output "dns1" {
  value = data.vcd_network_isolated.net.dns1
}

output "dhcp_start_address" {
  value = tolist(data.vcd_network_isolated.net.dhcp_pool)[0].start_address
}

output "dhcp_end_address" {
  value = tolist(data.vcd_network_isolated.net.dhcp_pool)[0].end_address
}

output "static_ip_start_address" {
  value = tolist(data.vcd_network_isolated.net.static_ip_pool)[0].start_address
}

output "static_ip_end_address" {
  value = tolist(data.vcd_network_isolated.net.static_ip_pool)[0].end_address
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `name` - (Required) A unique name for the network

» Attribute reference

All attributes defined in isolated network resource are supported.

» vcd_network_direct

Provides a vCloud Director Org VDC Network data source directly connected to an external network. This can be used to reference internal networks for vApps to connect.

Supported in provider *v2.5+*

» Example Usage

```
data "vcd_network_direct" "net" {
  org  = "my-org"
  vdc  = "my-vdc"
  name = "my-net"
}

# Get the name of the external network from the data source
# and use it to establish a second data source
output "external_network" {
  value = data.vcd_network_direct.net.external_network
}

data "vcd_external_network" "external_network1" {
  name = "${data.vcd_network_direct.net.external_network}"
}

# From the second data source we extract the basic networking info
output "gateway" {
  value = data.vcd_external_network.external_network1.ip_scope.0.gateway
}
# equivalent to
output "external_network_gateway" {
  value = data.vcd_network_direct.net.external_network_gateway
}

output "netmask" {
  value = data.vcd_external_network.external_network1.ip_scope.0.netmask
}
# equivalent to
output "external_network_netmask" {
  value = data.vcd_network_direct.net.external_network_netmask
}

output "DNS" {
  value = data.vcd_external_network.external_network1.ip_scope.0.dns1
}
# equivalent to
output "external_network_dns" {
  value = data.vcd_network_direct.net.external_network_dns1
}

output "external_ip" {
  value = data.vcd_external_network.external_network1.ip_scope.0.static_ip_pool.0.start_addr
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level.
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level.
- `name` - (Required) A unique name for the network

» Attribute Reference

- `external_network` - The name of the external network.
- `shared` - Defines if this network is shared between multiple vDCs in the vOrg.

» vcd_vapp

Provides a vCloud Director vApp data source. This can be used to reference vApps.

Supported in provider *v2.5+*

» Example Usage

```
data "vcd_vapp" "test-tf" {
  name             = "test-tf"
  org              = "tf"
  vdc              = "vdc-tf"
}

output "name" {
  value = data.vcd_vapp.test-tf.name
}

output "description" {
  value = data.vcd_vapp.test-tf.description
}

output "href" {
  value = data.vcd_vapp.test-tf.href
}

output "status_text" {
  value = data.vcd_vapp.test-tf.status_text
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) A unique name for the vApp
- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level

» Attribute reference

- `href` - The vApp Hyper Reference
- `metadata` - Key value map of metadata to assign to this vApp. Key and value can be any string.
- `guest_properties` - Key value map of vApp guest properties.
- `status` - The vApp status as a numeric code
- `status_text` - The vApp status as text.

» vcd_vapp_vm

Provides a vCloud Director VM data source. This can be used to access VMs within a vApp.

Supported in provider *v2.6+*

» Example Usage

```
data "vcd_vapp" "web" {
  name= "web"
}

data "vcd_vapp_vm" "web1" {
  vapp_name     = "${vcd_vapp.web.name}"
  name          = "web1"
}

output "vm" {
  value = data.vcd_vapp_vm.web1
}
```

Sample output: `vm = {   "computer_name" = "TestVM"   "cpu_cores" = 1   "cpus" = 2   "description" = "This OVA provides a minimal installed profile of PhotonOS. Default password for root user is changeme"   "disk" = []   "guest_properties" = {}   "href" = "https://my-vcd.org/api/vApp/vm-ecb449a2-0b11-494d-bbc7-6ae2f2ff9b82" "id" = "urn:vcloud:vm:ecb449a2-0b11-494d-bbc7-6ae2f2ff9b82" "memory" = 1024   "metadata" = {   "vm_metadata" = "VM Metadata." }   "name" = "vm-datacloud"   "network" = [{   "ip" = "192.168.2.10"   "ip_allocation_mode" = "MANUAL"   "is_primary" = true   "mac" = "00:50:56:29:08:89"   "name" = "net-datacloud-r" "type" = "org"   },   ]   "org" = "datacloud"   "storage_profile" = "*"   "vapp_name" = "vapp-datacloud"   "vdc" = "vdc-datacloud" }`

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `vapp_name` - (Required) The vApp this VM belongs to.
- `name` - (Required) A name for the VM, unique within the vApp

» Attribute reference

- `computer_name` - Computer name to assign to this virtual machine.
- `catalog_name` - The catalog name in which to find the given vApp Template
- `template_name` - The name of the vApp Template to use
- `memory` - The amount of RAM (in MB) allocated to the VM
- `cpus` - The number of virtual CPUs allocated to the VM
- `cpu_cores` - The number of cores per socket
- `metadata` - Key value map of metadata assigned to this VM
- `disk` - Independent disk attachment configuration.
- `network` - A block defining a network interface. Multiple can be used.

- `guest_properties` - Key value map of guest properties
- `description` - The VM description. Note: description is read only. Currently, this field has the description of the OVA used to create the VM
- `expose_hardware_virtualization` - Expose hardware-assisted CPU virtualization to guest OS

See VM resource for more info about VM attributes.

» vcd_nsxv_ip_set

Provides a vCloud Director IP set data source. An IP set is a group of IP addresses that you can add as the source or destination in a firewall rule or in DHCP relay configuration.

Supported in provider *v2.6+*

» Example Usage

```
data "vcd_nsxv_ip_set" "ip-set DS" {
  org                = "my-org"
  vdc                = "my-org-vdc"

  name = "not-managed"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `name` - (Required) IP set name for identifying the exact IP set

» Attribute Reference

All the attributes defined in `vcd_nsxv_ip_set` resource are available.

» vcd_nsxv_dhcp_relay

Provides a vCloud Director Edge Gateway DHCP relay configuration data source. The DHCP relay capability provided by NSX in vCloud Director environment allows to leverage existing DHCP infrastructure from within vCloud Director environment without any interruption to the IP address management in existing DHCP infrastructure. DHCP messages are relayed from virtual machines to the designated DHCP servers in your physical DHCP infrastructure, which allows IP addresses controlled by the NSX software to continue to be in sync with IP addresses in the rest of your DHCP-controlled environments.

Supported in provider *v2.6+*

» Example Usage 1

```
data "vcd_nsxv_dhcp_relay" "relay_config" {
  org          = "my-org"
  vdc          = "my-org-vdc"
  edge_gateway = "my-edge-gw"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations.
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level.
- `edge_gateway` - (Required) The name of the edge gateway on which DHCP relay is to be configured.

» Attribute Reference

All the attributes defined in `vcd_nsxv_dhcp_relay` resource are available.

» vcd_org

Provides a vCloud Director Org resource. This can be used to create, update, and delete an organization. Requires system administrator privileges.

Supported in provider *v2.0+*

» Example Usage

```
provider "vcd" {
  user     = "${var.admin_user}"
  password = "${var.admin_password}"
  org      = "System"
  url      = "https://AcmeVcd/api"
}

resource "vcd_org" "my-org" {
  name             = "my-org"
  full_name        = "My organization"
  description      = "The pride of my work"
  is_enabled       = "true"
  delete_recursive = "true"
  delete_force     = "true"
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) Org name
- `full_name` - (Required) Org full name
- `delete_recursive` - (Required) - pass `delete_recursive`=true as query parameter to remove an organization or VDC and any objects it contains that are in a state that normally allows removal.
- `delete_force` - (Required) - pass `delete_force=true` and `delete_recursive=true` to remove an organization or VDC and any objects it contains, regardless of their state.
- `is_enabled` - (Optional) - True if this organization is enabled (allows login and all other operations). Default is `true`.
- `description` - (Optional) - Org description. Default is empty.
- `deployed_vm_quota` - (Optional) - Maximum number of virtual machines that can be deployed simultaneously by a member of this organization. Default is unlimited (0)
- `stored_vm_quota` - (Optional) - Maximum number of virtual machines in vApps or vApp templates that can be stored in an undeployed state by a member of this organization. Default is unlimited (0)
- `can_publish_catalogs` - (Optional) - True if this organization is allowed to share catalogs. Default is `true`.
- `delay_after_power_on_seconds` - (Optional) - Specifies this organization's default for virtual machine boot delay after power on. Default is `0`.

» Importing

Supported in provider *v2.5+*

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing Org can be imported into this resource via supplying the path for an Org. Since the Org is at the top of the vCD hierarchy, the path corresponds to the Org name. For example, using this structure, representing an existing Org that was **not** created using Terraform:

```
resource "vcd_org" "my-orgadmin" {
  name             = "my-org"
  full_name        = "guessing"
  delete_recursive = "true"
  delete_force     = "true"
}
```

You can import such organization into terraform state using this command

```
terraform import vcd_org.my-org my-org
```

The state (in `terraform.tfstate`) would look like this:

```
{
  "version": 4,
  "terraform_version": "0.12.0",
  "serial": 1,
  "lineage": "4f328a1d-3ac3-a1be-b739-c1edde689335",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "vcd_org",
      "name": "my-org",
      "provider": "provider.vcd",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "can_publish_catalogs": true,
            "delay_after_power_on_seconds": null,
            "delete_force": null,
            "delete_recursive": null,
            "deployed_vm_quota": 50,
            "description": "",
            "full_name": "my-org",
            "id": "urn:vcloud:org:875e81c4-3d7a-4bf4-b7db-9d0abe0f0b0d",
            "is_enabled": true,
            "name": "my-org",
            "stored_vm_quota": 50
          }
        }
      ]
    }
  ]
}
```

After that, you can expand the configuration file and either update or delete the org as needed. Running `terraform plan` at this stage will show the difference between the minimal configuration file and the Org's stored properties.

» Sources

- OrgType
- ReferenceType
- Org deletion

» vcd_org_user

Provides a vCloud Director Org User. This can be used to create, update, and delete organization users, including org administrators.

Supported in provider *v2.4+*

Note: Only `System Administrator` or `Org Administrator` users can create users.

» Example Usage

```
# A simple user created with the minimum of properties
# uses the "password" field
resource "vcd_org_user" "my-org-admin" {
  org = "my-org"

  name          = "my-org-admin"
  description   = "a new org admin"
  role          = "Organization Administrator"
  password      = "change-me"
}

# Another user, created by filling all the fields
# Uses the "password_file" field.
resource "vcd_org_user" "test_user_vapp_author" {
  org = "datacloud"

  name              = "test_user_vapp_author"
  password_file     = "pwd201907101300.txt"
  full_name         = "test user vapp author"
  description       = "Org user test_user_vapp_author"
  role              = "vApp Author"
  enabled           = true
  take_ownership    = true
  provider_type     = "INTEGRATED"
  stored_vm_quota   = 20
  deployed_vm_quota = 20
  instant_messaging = "@test_user_vapp_author"
  email_address     = "test_user_vapp_author@test.company.org"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to which the VDC belongs. Optional if defined at provider level.
- `name` - (Required) A unique name for the user.
- `password` - (Optional, but required if `password_file` was not given) The user password. This value is never returned on read. It is inspected on create and modify. To modify, fill with a different value. Note that if you remove the password *on update*, Terraform will indicate that a change was occurring, but the empty password will be ignored by vCD.
- `password_file` (Optional, but required if `password` was not given). A text file containing the password. Recommended usage: after changing the password, run an apply again with the password blank. Using this property instead of `password` has the advantage that the sensitive data is not saved into Terraform state file. The disadvantage is that a password change requires also changing the file name.
- `provider_type` - (Optional) Identity provider type for this this user. One of: `INTEGRATED`, `SAML`, `OAUTH`. The default is `INTEGRATED`.
- `role` - (Required) The role of the user. Role names can be retrieved from the organization. Both built-in roles and custom built can be used. The roles normally available are:
  - `Organization Administrator`
  - `Catalog Author`
  - `vApp Author`
  - `vApp User`

    - `Console Access Only`
    - `Defer to Identity Provider`
- `full_name` - (Optional) The full name of the user.
- `description` - (Optional) An optional description of the user.
- `telephone` - (Optional) The Org User telephone number.
- `email_address` - (Optional) The Org User email address. Needs to be a properly formatted email address.
- `instant_messaging` - (Optional) The Org User instant messaging.
- `enabled` - (Optional) True if the user is enabled and can log in. The default is `true`.
- `is_group_role` - (Optional) True if this user has a group role.. The default is `false`.
- `is_locked` - (Optional)aIf the user account has been locked due to too many invalid login attempts, the value will change to true (only the system can lock the user). To unlock the user re-set this flag to false.
- `take_ownership` - (Optional) Take ownership of user's objects on deletion.
- `deployed_vm_quota` - (Optional) Quota of vApps that this user can deploy. A value of 0 specifies an unlimited quota. The default is 10.
- `stored_vm_quota` - (Optional) Quota of vApps that this user can store. A value of 0 specifies an unlimited quota. The default is 10.

» Attribute Reference

The following attributes are exported on this resource:

- `id` - The ID of the Organization user

» Importing

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing user can be imported into this resource via supplying the full dot separated path for an org user. For example, using this structure, representing an existing user that was **not** created using Terraform:

```
resource "vcd_org_user" "my-org-admin" {
  org  = "my-org"
  name = "my-org-admin"
  role = "Organization Administrator"
}
```

You can import such user into terraform state using this command

```
terraform import vcd_org_user.my-org-admin my-org.my-org-admin
```

NOTE: the default separator (.) can be changed using Provider.import_separator or variable VCD_IMPORT_SEPARATOR

The state (in `terraform.tfstate`) would look like this:

```
{
  "version": 4,
  "terraform_version": "0.12.0",
  "serial": 1,
  "lineage": "f3fb8d07-8fe5-4fe3-3afe-c9050ffe68f6",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "vcd_org_user",
      "name": "my-org-user",
      "provider": "provider.vcd",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "deployed_vm_quota": 50,
            "description": "This is my-org main user",
            "email_address": "my-org-admin@mycompany.com",
            "full_name": "My Org Admin",
            "id": "urn:vcloud:user:5fd69dfa-6bbe-40a6-9ee3-70448b6601ef",
            "instant_messaging": "@my_org_admin",
            "enabled": true,
            "is_group_role": false,
            "is_locked": false,
            "name": "my-org-user",
            "org": "my-org",
            "password": null,
            "password_file": null,
            "provider_type": "INTEGRATED",
            "role": "Organization Administrator",
            "stored_vm_quota": 50,
            "take_ownership": null,
            "telephone": "123-456-7890"
          }
        }
      ]
    }
  ]
}
```

After that, you can expand the configuration file and either update or delete the user as needed. Running `terraform plan` at this stage will show the difference between the minimal configuration file and the user's stored properties.

» vcd_catalog

Provides a vCloud Director catalog resource. This can be used to create and delete a catalog.

Supported in provider *v2.0+*

» Example Usage

```
resource "vcd_catalog" "myNewCatalog" {
  org = "my-org"

  name             = "my-catalog"
  description      = "catalog for files"
  delete_recursive = "true"
  delete_force     = "true"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `name` - (Required) Catalog name
- `description` - (Optional) - Description of catalog
- `delete_recursive` - (Required) - When destroying use delete_recursive=True to remove the catalog and any objects it contains that are in a state that normally allows removal
- `delete_force` -(Required) - When destroying use delete_force=True with delete_recursive=True to remove a catalog and any objects it contains, regardless of their state

» Importing

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing catalog can be imported into this resource via supplying the full dot separated path for a catalog. For example, using this structure, representing an existing catalog that was **not** created using Terraform:

```
resource "vcd_catalog" "my-catalog" {
  org              = "my-org"
  name             = "my-catalog"
  delete_recursive = "true"
  delete_force     = "true"
}
```

You can import such catalog into terraform state using this command

```
terraform import vcd_catalog.my-catalog my-org.my-catalog
```

NOTE: the default separator (.) can be changed using Provider.import_separator or variable VCD_IMPORT_SEPARATOR

After that, you can expand the configuration file and either update or delete the catalog as needed. Running `terraform plan` at this stage will show the difference between the minimal configuration file and the catalog's stored properties.

» vcd_catalog_item

Provides a vCloud Director catalog item resource. This can be used to upload OVA to catalog and delete it.

Supported in provider *v2.0+*

» Example Usage

```
resource "vcd_catalog_item" "myNewCatalogItem" {
  org     = "my-org"
  catalog = "my-catalog"

  name                 = "my ova"
  description          = "new vapp template"
  ova_path             = "/home/user/file.ova"
  upload_piece_size    = 10
  show_upload_progress = true

  metadata = {
    license = "public"
    version = "v1"
  }
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `catalog` - (Required) The name of the catalog where to upload OVA file
- `name` - (Required) Item name in catalog
- `description` - (Optional) - Description of item
- `ova_path` - (Required) - Absolute or relative path to file to upload
- `upload_piece_size` - (Optional) - Size in MB for splitting upload size. It can possibly impact upload performance. Default 1MB.
- `show_upload_progress` - (Optional) - Default false. Allows to see upload progress
- `metadata` - (Optional; *v2.5+*) Key value map of metadata to assign

» Importing

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing catalog item can be imported into this resource via supplying the full dot separated path for a catalog item. For example, using this structure, representing an existing catalog item that was **not** created using Terraform:

```
resource "vcd_catalog_item" "my-item" {
  org         = "my-org"
  catalog     = "my-catalog"
  name        = "my-item"
  ova_path    = "guess"
}
```

You can import such catalog item into terraform state using this command

```
terraform import vcd_catalog_item.my-item my-org.my-catalog.my-item
```

NOTE: the default separator (.) can be changed using Provider.import_separator or variable VCD_IMPORT_SEPARATOR

After that, you can expand the configuration file and either update or delete the catalog item as needed. Running `terraform plan` at this stage will show the difference between the minimal configuration file and the item's stored properties.

» vcd_catalog_media

Provides a vCloud Director media resource. This can be used to upload media to catalog and delete it.

Supported in provider *v2.0+*

» Example Usage

```
resource "vcd_catalog_media" "myNewMedia" {
  org     = "my-org"
  catalog = "my-catalog"

  name                 = "my iso"
  description          = "new os versions"
  media_path           = "/home/user/file.iso"
  upload_piece_size    = 10
  show_upload_progress = true

  metadata = {
    license = "public"
    version = "v1"
  }

}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `catalog` - (Required) The name of the catalog where to upload media file
- `name` - (Required) Media file name in catalog
- `description` - (Optional) - Description of media file
- `media_path` - (Required) - Absolute or relative path to file to upload
- `upload_piece_size` - (Optional) - size in MB for splitting upload size. It can possibly impact upload performance. Default 1MB.
- `show_upload_progress` - (Optional) - Default false. Allows to see upload progress
- `metadata` - (Optional; *v2.5+*) Key value map of metadata to assign

» Attribute reference

Supported in provider *v2.5+*

- `is_iso` - (Computed) returns True if this media file is ISO
- `owner_name` - (Computed) returns owner name
- `is_published` - (Computed) returns True if this media file is in a published catalog
- `creation_date` - (Computed) returns creation date
- `size` - (Computed) returns media storage in Bytes
- `status` - (Computed) returns media status
- `storage_profile_name` - (Computed) returns storage profile name

» Importing

Supported in provider *v2.5+*

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing media item can be imported into this resource via supplying its path. The path for this resource is made of org-name.catalog-name.media-name For example, using this structure, representing a media item that was **not** created using Terraform:

```
resource "vcd_catalog_media" "tf-mymedia" {
  org     = "my-org"
  catalog = "my-catalog"
  name    = "my-media"
}
```

You can import such catalog media into terraform state using this command

```
terraform import vcd_catalog_media.tf-mymedia my-org.my-catalog.my-media
```

After importing, if you run `terraform plan` you will see the rest of the values and modify the script accordingly for further operations.

» vcd_dnat

Provides a vCloud Director DNAT resource. This can be used to create, modify, and delete destination NATs to map an external IP/port to an internal IP/port.

Note: This resource may corrupt UI edited NAT rules when used with advanced edge gateways. Please use `vcd_nsxv_dnat` in that case.

Note: From v2.4+ `protocol` requires lower case values. This may result in invalid configuration if upper case was used previously.

Warning: When advanced edge gateway is used and the rule is updated using UI, then ID mapping will be lost and Terraform won't find the rule anymore and remove it from state.

» Example Usage

```
resource "vcd_dnat" "web" {
  org = "my-org" # Optional
  vdc = "my-vdc" # Optional

  edge_gateway    = "Edge Gateway Name"
  external_ip     = "78.101.10.20"
  port            = 80
  internal_ip     = "10.10.0.5"
  translated_port = 8080
}

resource "vcd_dnat" "forIcmp" {
  org = "my-org" # Optional
  vdc = "my-vdc" # Optional

  network_name = "my-external-network"
  network_type = "ext"

  edge_gateway  = "Edge Gateway Name"
  external_ip   = "78.101.10.20"
  port          = -1                    # "-1" == "any"
  internal_ip   = "10.10.0.5"
  protocol      = "icmp"
  icmp_sub_type = "router-solicitation"
}
```

» Argument Reference

The following arguments are supported:

- `edge_gateway` - (Required) The name of the edge gateway on which to apply the DNAT
- `external_ip` - (Required) One of the external IPs available on your Edge Gateway
- `port` - (Required) The port number to map. -1 translates to ”any”
- `translated_port` - (Optional) The port number to map
- `internal_ip` - (Required) The IP of the VM to map to

- `protocol` - (Optional; *v2.0+*) The protocol type. Possible values are `tcp`, `udp`, `tcpupd`, `icmp`, `any`. `tcp` is default to be backward compatible with previous version
- `icmp_sub_type` - (Optional; *v2.0+*) The name of ICMP type. Possible values are address-mask-request, destination-unreachable, echo-request, echo-reply, parameter-problem, redirect, router-advertisement, router-solicitation, source-quench, time-exceeded, timestamp-request, timestamp-reply, any
- `network_type` - (Optional; *v2.4+*) Type of the network on which to apply the NAT rule. Possible values `org` or `ext`. `ext` requires system administrator privileges. *`network_type` will be a required field in the next major version.*
- `network_name` - (Optional; *v2.4+*) The name of the network on which to apply the SNAT. *`network_name` will be a required field in the next major version.*
- `org` - (Optional; *v2.0+*) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional; *v2.0+*) The name of VDC to use, optional if defined at provider level
- `description` - (Optional; *v2.4+*) - Description of item

» vcd_external_network

Provides a vCloud Director external network resource. This can be used to create and delete external networks. Requires system administrator privileges.

Supported in provider *v2.2+*

» Example Usage

```
provider "vcd" {
  user     = "${var.admin_user}"
  password = "${var.admin_password}"
  org      = "System"
  url      = "https://Vcd/api"
}

resource "vcd_external_network" "net" {
  name        = "my-ext-net"
  description = "Reference for vCD external network"

  ip_scope {
    gateway    = "192.168.30.49"
    netmask    = "255.255.255.240"
    dns1       = "192.168.0.164"
    dns2       = "192.168.0.196"
    dns_suffix = "mybiz.biz"

    static_ip_pool {
      start_address = "192.168.30.51"
      end_address   = "192.168.30.62"
    }
  }

  # It's possible to define more than one IP scope
  ip_scope {
    gateway    = "192.168.31.49"
    netmask    = "255.255.255.240"
    dns1       = "192.168.1.164"
    dns2       = "192.168.1.196"
    dns_suffix = "my.biz"

    static_ip_pool {
      start_address = "192.168.31.51"
      end_address   = "192.168.31.55"
    }

    static_ip_pool {
      start_address = "192.168.31.57"
      end_address   = "192.168.31.59"
    }
  }

  vsphere_network {
    name    = "myNetwork"
    type    = "DV_PORTGROUP"
    vcenter = "vcenter-name"
  }

  # It's possible to define more than one vSphere network
  vsphere_network {
    name    = "myNetwork2"
    type    = "DV_PORTGROUP"
    vcenter = "vcenter-name2"
  }

  retain_net_info_across_deployments = "false"
}
```

```
resource "vcd_network_direct" "net" {
  org              = "my-org"
  vdc              = "my-vdc"
  name             = "my-net"
  external_network = "${vcd_external_network.net.name}"
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) A unique name for the network
- `description` - (Optional) Network friendly description
- `ip_scope` - (Required) A list of IP scopes for the network. See IP Scope below for details.
- `vsphere_network` - (Required) A list of DV_PORTGROUP or NETWORK objects names that back this network. Each referenced DV_PORTGROUP or NETWORK must exist on a vCenter server registered with the system. See vSphere Network below for details.
- `retain_net_info_across_deployments` - (Optional) Specifies whether the network resources such as IP/MAC of router will be retained across deployments. Default is false.

» IP Scope

- `gateway` - (Required) Gateway of the network
- `netmask` - (Required) Network mask
- `dns1` - (Optional) Primary DNS server
- `dns2` - (Optional) Secondary DNS server
- `dns_suffix` (Optional) A FQDN for the virtual machines on this network.
- `static_ip_pool` - (Required) IP ranges used for static pool allocation in the network. See IP Pool below for details.

» IP Pool

- `start_address` - (Required) Start address of the IP range
- `end_address` - (Required) End address of the IP range

» vSphere Network

- `name` - (Required) Port group name

- `type` - (Required) The vSphere type of the object. One of: DV_PORTGROUP (distributed virtual port group), NETWORK (standard switch port group)
- `vcenter` - (Required) The vCenter server name

» Importing

Supported in provider *v2.5+*

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing external network can be imported into this resource via supplying the path for an external network. Since the external network is at the top of the vCD hierarchy, the path corresponds to the external network name. For example, using this structure, representing an existing external network that was **not** created using Terraform:

```
resource "vcd_external_network" "tf-external-network" {
  name             = "my-ext-net"
}
```

You can import such external network into terraform state using this command

```
terraform import vcd_external_network.tf-external-network my-ext-net
```

NOTE: the default separator (.) can be changed using Provider.import_separator or variable VCD_IMPORT_SEPARATOR

While the above structure is the minimum needed to get an import, it is not sufficient to run `terraform plan`, as it lacks several mandatory fields. To use the imported resource, you will need to add the missing properties using the data in `terraform.tfstate` as a reference. If the resource does not need modifications, consider using an external network data source instead.

» vcd_firewall_rules

Provides a vCloud Director Firewall resource. This can be used to create, modify, and delete firewall settings and rules.

Note: Please use the improved `vcd_nsxv_firewall_rule` resource with advanced edge gateways (NSX-V).

Note: Using this resource automatically enables default firewall rule logging. This may cause `vcd_edgegateway` resource to report changes for field `fw_default_rule_logging_enabled` during `plan`/`apply` phases.

» Example Usage

```
resource "vcd_firewall_rules" "fw" {
  edge_gateway   = "Edge Gateway Name"
  default_action = "drop"

  rule {
    description      = "drop-ftp-out"
    policy           = "drop"
    protocol         = "tcp"
    destination_port = "21"
    destination_ip   = "any"
    source_port      = "any"
    source_ip        = "10.10.0.0/24"
  }

  rule {
    description      = "allow-outbound"
    policy           = "allow"
    protocol         = "any"
    destination_port = "any"
    destination_ip   = "any"
    source_port      = "any"
    source_ip        = "10.10.0.0/24"
  }
}

resource "vcd_vapp" "web" {
  # ...
}

resource "vcd_firewall_rules" "fw-web" {
  edge_gateway   = "Edge Gateway Name"
  default_action = "drop"

  rule {
    description      = "allow-web"
    policy           = "allow"
    protocol         = "tcp"
    destination_port = "80"
    destination_ip   = "${vcd_vapp.web.ip}"
    source_port      = "any"
    source_ip        = "any"
  }
}
```

» Argument Reference

The following arguments are supported:

- edge_gateway - (Required) The name of the edge gateway on which to apply the Firewall Rules
- default_action - (Required) Either ”allow” or ”drop”. Specifies what to do should none of the rules match
- rule - (Optional) Configures a firewall rule; see Rules below for details.
- org - (Optional; *v2.0+*) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- vdc - (Optional; *v2.0+*) The name of VDC to use, optional if defined at provider level

» Rules

Each firewall rule supports the following attributes:

- description - (Required) Description of the fireall rule
- policy - (Required) Specifies what to do when this rule is matched. Either ”allow” or ”drop”
- protocol - (Required) The protocol to match. One of ”tcp”, ”udp”, ”icmp” or ”any”
- destination_port - (Required) The destination port to match. Either a port number or ”any”
- destination_ip - (Required) The destination IP to match. Either an IP address, IP range or ”any”
- source_port - (Required) The source port to match. Either a port number or ”any”
- source_ip - (Required) The source IP to match. Either an IP address, IP range or ”any”

» vcd_independent_disk

Provides a vCloud Director independent disk resource. This can be used to create and delete independent disks.

Supported in provider *v2.1+*

» Example Usage

```
resource "vcd_independent_disk" "myNewIndependentDisk" {
  vdc             = "my-vcd"
  name            = "logDisk"
  size            = "33000"
  bus_type        = "SCSI"
  bus_sub_type    = "VirtualSCSI"
  storage_profile = "external"
}

resource "vcd_vapp_vm" "web2" {
  vapp_name     = "${vcd_vapp.web.name}"

...

  disk {
    name = "${vcd_independent_disk.myNewIndependentDisk.name}"
    bus_number = 1
    unit_number = 0
  }

  depends_on = ["vcd_independent_disk.myNewIndependentDisk"]
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `name` - (Required) Disk name
- `size` - (Required) Size of disk in MB. On read this values isn't refreshed.
- `bus_type` - (Optional) Disk bus type. Values can be: `IDE`, `SCSI`, `SATA`
- `bus_sub_type` - (Optional) Disk bus subtype. Values can be: `buslogic`, `lsilogic`, `lsilogicsas`, `VirtualSCSI` for `SCSI` and `ahci` for `SATA`
- `storage_profile` - (Optional) The name of storage profile where disk will be created

» Attribute reference

Supported in provider *v2.5+*

- `iops` - (Computed) IOPS request for the created disk
- `owner_name` - (Computed) The owner name of the disk

- `datastore_name` - (Computed) Data store name. Readable only for system user.
- `is_attached` - (Computed) True if the disk is already attached

» Importing

Supported in provider *v2.5+*

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing independent disk can be imported into this resource via supplying its path. The path for this resource is made of org-name.vdc-name.disk-id For example, using this structure, representing a independent disk that was **not** created using Terraform:

```
resource "vcd_independent_disk" "tf-myDisk" {
  vdc     = "my-vdc"
  name    = "my-disk"
}
```

You can import such independent disk into terraform state using this command

```
terraform import vcd_independent_disk.tf-myDisk org-name.vdc-name.my-disk-id
```

After importing, if you run `terraform plan` you will see the rest of the values and modify the script accordingly for further operations.

» Listing independent disk IDs

If you want to list IDs there is a special command **`terraform import vcd_independent_disk.imported list@org-name.vdc-name.my-independent-disk-name`** where `org-name` is the organization used, `vdc-name` is vDC name and `my-independent-disk-name` is independent disk name. The output for this command should look similar to below one:

```
$ terraform import vcd_independent_disk.imported list@org-name.vdc-name.my-independent-disk-
vcd_independent_disk.Disk_import: Importing from ID "list@org-name.vdc-name.my-independent-d
Retrieving all disks by name
No  ID                                                     Name    Description Size
--  --                                                     ----    ------      ----
1  urn:vcloud:disk:1bbc273d-7701-4f06-97be-428b46b0805e     diskV2  loging      78946548
2  urn:vcloud:disk:6e1c996f-48b8-4e78-8111-a6407188d8b6     diskV2              5557452

Error: resource was not imported! resource id must be specified in one of these formats:
'org-name.vdc-name.my-independent-disk-id' to import by rule id
'list@org-name.vdc-name.my-independent-disk-name' to get a list of disks with their IDs
```

Now to import disk with ID urn:vcloud:disk:1bbc273d-7701-4f06-97be-428b46b0805e one could supply this command:

```
$ terraform import vcd_independent_disk.imported list@org-name.vdc-name.urn:vcloud:disk:1bbc
```

» vcd_inserted_media

Provides a vCloud Director resource for inserting or ejecting media (ISO) file for the VM. Create this resource for inserting the media, and destroy it for ejecting.

Supported in provider *v2.0+*

» Example Usage

```
resource "vcd_inserted_media" "myInsertedMedia" {
  org     = "my-org"
  vdc     = "my-vcd"
  catalog = "my-catalog"
  name    = "my-iso"

  vapp_name = "my-vApp"
  vm_name   = "my-VM"

  eject_force = true
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional; *v2.0+*) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional; *v2.0+*) The name of VDC to use, optional if defined at provider level
- `catalog` - (Required) The name of the catalog where to find media file
- `name` - (Required) Media file name in catalog which will be inserted to VM
- `vapp_name` - (Required) - The name of vApp to find
- `vm_name` - (Required) - The name of VM to be used to insert media file
- `eject_force` - (Optional; *v2.1+*) Allows to pass answer to question in vCD ”The guest operating system has locked the CD-ROM door and is probably using the CD-ROM. Disconnect anyway (and override the lock)?” when ejecting from a VM which is powered on. True means ”Yes” as answer to question. Default is `true`

» vcd_network (Deprecated)

Provides a vCloud Director Org VDC Network. This can be used to create, modify, and delete internal networks for vApps to connect.

Deprecated in v2.0+ : this resource is deprecated and replaced by vcd-network-routed. It is also complemented by vcd-network-isolated and vcd-network-direct.

» Example Usage

```
resource "vcd_network" "net" {
  name         = "my-net"
  edge_gateway = "Edge Gateway Name"
  gateway      = "10.10.0.1"

  dhcp_pool {
    start_address = "10.10.0.2"
    end_address   = "10.10.0.100"
  }

  static_ip_pool {
    start_address = "10.10.0.152"
    end_address   = "10.10.0.254"
  }
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) A unique name for the network
- `edge_gateway` - (Required) The name of the edge gateway
- `netmask` - (Optional) The netmask for the new network. Defaults to `255.255.255.0`
- `gateway` (Required) The gateway for this network
- `dns1` - (Optional) First DNS server to use. Defaults to `8.8.8.8`
- `dns2` - (Optional) Second DNS server to use. Defaults to `8.8.4.4`
- `dns_suffix` - (Optional) A FQDN for the virtual machines on this network
- `shared` - (Optional) Defines if this network is shared between multiple vDCs in the vOrg. Defaults to `false`.
- `dhcp_pool` - (Optional) A range of IPs to issue to virtual machines that don't have a static IP; see IP Pools below for details.

- `static_ip_pool` - (Optional) A range of IPs permitted to be used as static IPs for virtual machines; see IP Pools below for details.

» IP Pools

Static IP Pools and DHCP Pools support the following attributes:

- `start_address` - (Required) The first address in the IP Range
- `end_address` - (Required) The final address in the IP Range

DHCP Pools additionally support the following attributes:

- `default_lease_time` - (Optional) The default DHCP lease time to use. Defaults to `3600`.
- `max_lease_time` - (Optional) The maximum DHCP lease time to use. Defaults to `7200`.

» vcd_network_routed

Provides a vCloud Director Org VDC routed Network. This can be used to create, modify, and delete internal networks for vApps to connect.

Supported in provider *v2.0+*

» Example Usage

```
resource "vcd_network_routed" "net" {
  org = "my-org" # Optional
  vdc = "my-vdc" # Optional

  name         = "my-net"
  edge_gateway = "Edge Gateway Name"
  gateway      = "10.10.0.1"

  dhcp_pool {
    start_address = "10.10.0.2"
    end_address   = "10.10.0.100"
  }

  static_ip_pool {
    start_address = "10.10.0.152"
    end_address   = "10.10.0.254"
  }
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional; *v2.0+*) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional; *v2.0+*) The name of VDC to use, optional if defined at provider level
- `name` - (Required) A unique name for the network
- `description` - (Optional *v2.6+*) An optional description of the network
- `interface_type` - (Optional *v2.6+*) An interface for the network. One of `internal` (default), `subinterface`, `distributed` (requires the edge gateway to support distributed networks)
- `edge_gateway` - (Required) The name of the edge gateway
- `netmask` - (Optional) The netmask for the new network. Defaults to `255.255.255.0`
- `gateway` (Required) The gateway for this network
- `dns1` - (Optional) First DNS server to use. Defaults to `8.8.8.8`
- `dns2` - (Optional) Second DNS server to use. Defaults to `8.8.4.4`
- `dns_suffix` - (Optional) A FQDN for the virtual machines on this network
- `shared` - (Optional) Defines if this network is shared between multiple VDCs in the Org. Defaults to `false`.
- `dhcp_pool` - (Optional) A range of IPs to issue to virtual machines that don't have a static IP; see IP Pools below for details.
- `static_ip_pool` - (Optional) A range of IPs permitted to be used as static IPs for virtual machines; see IP Pools below for details.

» IP Pools

Static IP Pools and DHCP Pools support the following attributes:

- `start_address` - (Required) The first address in the IP Range
- `end_address` - (Required) The final address in the IP Range

DHCP Pools additionally support the following attributes:

- `default_lease_time` - (Optional) The default DHCP lease time to use. Defaults to `3600`.
- `max_lease_time` - (Optional) The maximum DHCP lease time to use. Defaults to `7200`.

» Importing

Supported in provider *v2.5+*

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing routed network can be imported into this resource via supplying its path. The path for this resource is made of orgName.vdcName.networkName. For example, using this structure, representing a routed network that was **not** created using Terraform:

```
resource "vcd_network_routed" "tf-mynet" {
  name              = "my-net"
  org               = "my-org"
  vdc               = "my-vdc"
  edge_gateway      = "COMPUTE"
  gateway           = "COMPUTE"
}
```

You can import such routed network into terraform state using this command

```
terraform import vcd_network_routed.tf-mynet my-org.my-vdc.my-net
```

NOTE: the default separator (.) can be changed using Provider.import_separator or variable VCD_IMPORT_SEPARATOR

After importing, if you run `terraform plan` you will see the rest of the values and modify the script accordingly for further operations.

» vcd_network_direct

Provides a vCloud Director Org VDC Network directly connected to an external network. This can be used to create, modify, and delete internal networks for vApps to connect.

Supported in provider *v2.0+*

Note: Only `System Administrator` can create an organization virtual datacenter network that connects directly to an external network. You must use `System Adminstrator` account in `provider` configuration and then provide `org` and `vdc` arguments for direct networks to work.

» Example Usage

```
resource "vcd_network_direct" "net" {
  org = "my-org" # Optional
  vdc = "my-vdc" # Optional

  name             = "my-net"
  external_network = "my-ext-net"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional; *v2.0+*) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional; *v2.0+*) The name of VDC to use, optional if defined at provider level
- `name` - (Required) A unique name for the network
- `description` - (Optional *v2.6+*) An optional description of the network
- `external_network` - (Required) The name of the external network.
- `shared` - (Optional) Defines if this network is shared between multiple VDCs in the Org. Defaults to `false`.

» Attribute reference

Supported in provider *v2.5+*

- `external_network_gateway` - (Computed) returns the gateway from the external network
- `external_network_netmask` - (Computed) returns the netmask from the external network
- `external_network_dns1` - (Computed) returns the first DNS from the external network
- `external_network_dns2` - (Computed) returns the second DNS from the external network
- `external_network_dns_suffix` - (Computed) returns the DNS suffix from the external network

» Importing

Supported in provider *v2.5+*

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing direct network can be imported into this resource via supplying its path. The path for this resource is made of orgName.vdcName.networkName. For example, using this structure, representing a direct network that was **not** created using Terraform:

```
resource "vcd_network_direct" "tf-mynet" {
  name              = "my-net"
  org               = "my-org"
  vdc               = "my-vdc"
  external_network  = "COMPUTE"
}
```

You can import such isolated network into terraform state using this command

```
terraform import vcd_network_direct.tf-mynet my-org.my-vdc.my-net
```

NOTE: the default separator (.) can be changed using Provider.import_separator or variable VCD_IMPORT_SEPARATOR

After importing, if you run `terraform plan` you will see the rest of the values and modify the script accordingly for further operations.

» vcd_network_isolated

Provides a vCloud Director Org VDC isolated Network. This can be used to create, modify, and delete internal networks for vApps to connect. This network is not attached to external networks or routers.

Supported in provider *v2.0+*

» Example Usage

```
resource "vcd_network_isolated" "net" {
  org = "my-org" # Optional
  vdc = "my-vdc" # Optional

  name    = "my-net"
  gateway = "192.168.2.1"
  dns1    = "192.168.2.1"

  dhcp_pool {
    start_address = "192.168.2.2"
    end_address   = "192.168.2.50"
  }

  static_ip_pool {
    start_address = "192.168.2.51"
    end_address   = "192.168.2.100"
  }
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional; *v2.0+*) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional; *v2.0+*) The name of VDC to use, optional if defined at provider level
- `name` - (Required) A unique name for the network
- `description` - (Optional *v2.6+*) An optional description of the network
- `netmask` - (Optional) The netmask for the new network. Defaults to `255.255.255.0`
- `gateway` (Required) The gateway for this network
- `dns1` - (Optional) First DNS server to use. Defaults to `8.8.8.8`
- `dns2` - (Optional) Second DNS server to use. Defaults to `8.8.4.4`
- `dns_suffix` - (Optional) A FQDN for the virtual machines on this network
- `shared` - (Optional) Defines if this network is shared between multiple VDCs in the Org. Defaults to `false`.
- `dhcp_pool` - (Optional) A range of IPs to issue to virtual machines that don't have a static IP; see IP Pools below for details.
- `static_ip_pool` - (Optional) A range of IPs permitted to be used as static IPs for virtual machines; see IP Pools below for details.

» IP Pools

Static IP Pools and DHCP Pools support the following attributes:

- `start_address` - (Required) The first address in the IP Range
- `end_address` - (Required) The final address in the IP Range

DHCP Pools additionally support the following attributes:

- `default_lease_time` - (Optional) The default DHCP lease time to use. Defaults to `3600`.
- `max_lease_time` - (Optional) The maximum DHCP lease time to use. Defaults to `7200`.

» Importing

Supported in provider *v2.5+*

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing isolated network can be imported into this resource via supplying its path. The path for this resource is made of orgName.vdcName.networkName.

For example, using this structure, representing an isolated network that was **not** created using Terraform:

```
resource "vcd_network_isolated" "tf-mynet" {
  name              = "my-net"
  org               = "my-org"
  vdc               = "my-vdc"
  gateway           = "COMPUTE"
}
```

You can import such isolated network into terraform state using this command

```
terraform import vcd_network_isolated.tf-mynet my-org.my-vdc.my-net
```

NOTE: the default separator (.) can be changed using Provider.import_separator or variable VCD_IMPORT_SEPARATOR

After importing, if you run `terraform plan` you will see the rest of the values and modify the script accordingly for further operations.

» vcd_snat

Provides a vCloud Director SNAT resource. This can be used to create, modify, and delete source NATs to allow vApps to send external traffic.

Note: This resource may corrupt UI edited NAT rules when used with advanced edge gateways. Please use `vcd_nsxv_snat` in that case.

Warning: When advanced edge gateway is used and the rule is updated using UI, then ID mapping will be lost and Terraform won't find the rule anymore and remove it from state.

» Example Usage

```
resource "vcd_snat" "outbound" {
  edge_gateway = "Edge Gateway Name"
  network_name = "my-org-vdc-network"
  network_type = "org"
  external_ip  = "78.101.10.20"
  internal_ip  = "10.10.0.0/24"
}
```

» Argument Reference

The following arguments are supported:

- `edge_gateway` - (Required) The name of the edge gateway on which to apply the SNAT
- `external_ip` - (Required) One of the external IPs available on your Edge Gateway
- `internal_ip` - (Required) The IP or IP Range of the VM(s) to map from
- `network_type` - (Optional; *v2.4+*) Type of the network on which to apply the NAT rule. Possible values `org` or `ext`. *`network_type` will be a required field in the next major version.*
- `network_name` - (Optional; *v2.4+*) The name of the network on which to apply the SNAT. *`network_name` will be a required field in the next major version.*
- `org` - (Optional; *v2.0+*) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional; *v2.0+*) The name of VDC to use, optional if defined at provider level
- `description` - (Optional; *v2.4+*) - Description of item

» vcd_edgegateway

Provides a vCloud Director edge gateway directly connected to one or more external networks. This can be used to create and delete edge gateways for Org VDC networks to connect.

Supported in provider *v2.4+*

Note: Only `System Administrator` can create an edge gateway. You must use `System Adminstrator` account in `provider` configuration and then provide `org` and `vdc` arguments for edge gateway to work.

Note: Load balancing capabilities will work only when edge gateway is `advanced`. Load balancing settings will be **ignored** when it is not. Refer to official vCloud Director documentation for more information.

» Example Usage

```
resource "vcd_edgegateway" "egw" {
  org = "my-org"
  vdc = "my-vdc"

  name                    = "my-egw"
  description             = "new edge gateway"
  configuration           = "compact"
  advanced                = true
  external_network {
    name = "my-ext-net1"

    subnet {
      ip_address            = "192.168.30.51"
      gateway               = "192.168.30.49"
      netmask               = "255.255.255.240"
      use_for_default_route = true
    }
  }
}

resource "vcd_network_routed" "rnet1" {
  name         = "rnet1"
  org          = "my-org"
  vdc          = "my-vdc"
  edge_gateway = vcd_edgegateway.egw.name
  gateway      = "192.168.2.1"

  static_ip_pool {
    start_address = "192.168.2.2"
    end_address   = "192.168.2.100"
  }
}
```

» Example Usage (multiple External Networks, Subnets and IP pool sub-allocation)

```
resource "vcd_edgegateway" "egw" {
  org = "my-org"
  vdc = "my-vdc"

  name          = "edge-with-complex-networks"
  description   = "new edge gateway"
  configuration = "compact"
  advanced      = true


  external_network {
    name = "my-main-external-network"

    subnet {
      ip_address = "192.168.30.51"
      gateway    = "192.168.30.49"
      netmask    = "255.255.255.240"
      suballocate_pool {
        start_address = "192.168.30.53"
        end_address   = "192.168.30.55"
      }

      suballocate_pool {
        start_address = "192.168.30.58"
        end_address   = "192.168.30.60"
      }
    }

    subnet {
      # ip_address is skipped here on purpose to get dynamic IP assigned. Because this
      # subnet is used for default route, this IP address can then be accessed using
      # `default_external_network_ip` attribute.
      use_for_default_route = true
      gateway               = "192.168.40.149"
      netmask               = "255.255.255.0"
    }
  }

  external_network {
    name = "my-other-external-network"

    subnet {
      # IP address will be auto-assigned. It can then be found in the list of `external_netw
      # attribute
      gateway    = "1.1.1.1"
      netmask    = "255.255.255.248"
    }
  }
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to which the VDC belongs. Optional if defined at provider level.
- `vdc` - (Optional) The name of VDC that owns the edge gateway. Optional if defined at provider level.
- `name` - (Required) A unique name for the edge gateway.
- `external_networks` - (Deprecated, Optional) An array of external network names. This supports simple external networks with one subnet

only. **Please use** the external network block structure to define external networks.

- `external_network` - (Optional, *v2.6+*) One or more blocks defining external networks, their subnets, IP addresses and IP pool suballocation attached to edge gateway interfaces. Details are in external network block below.
- `configuration` - (Required) Configuration of the vShield edge VM for this gateway. One of: `compact`, `full` (”Large”), `x-large`, `full4` (”Quad Large”).
- `default_gateway_network` - (Deprecated, Optional) Name of the external network to be used as default gateway. It must be included in the list of `external_networks`. Providing an empty string or omitting the argument will create the edge gateway without a default gateway. **Please use** the external network block structure and `use_for_default_route` to specify a subnet which should be used as a default route.
- `advanced` - (Optional) True if the gateway uses advanced networking. Default is `true`.
- `ha_enabled` - (Optional) Enable high availability on this edge gateway. Default is `false`.
- `distributed_routing` - (Optional) If advanced networking enabled, also enable distributed routing. Default is `false`.
- `fips_mode_enabled` - (Optional) When FIPS mode is enabled, any secure communication to or from the NSX Edge uses cryptographic algorithms or protocols that are allowed by United States Federal Information Processing Standards (FIPS). FIPS mode turns on the cipher suites that comply with FIPS. Default is `false`. **Note:** to use FIPS mode it must be enabled in vCD system settings and is only supported starting with vCD version 9.1. This field **must not** be set for vCD 9.0.
- `use_default_route_for_dns_relay` - (Optional) When default route is set, it will be used for gateways’ default routing and DNS forwarding. Default is `false`.
- `lb_enabled` - (Optional) Enable load balancing. Default is `false`.
- `lb_acceleration_enabled` - (Optional) Enable to configure the load balancer to use the faster L4 engine rather than L7 engine. The L4 TCP VIP is processed before the edge gateway firewall so no `allow` firewall rule is required. Default is `false`. **Note:** L7 VIPs for HTTP and HTTPS are processed after the firewall, so when Acceleration Enabled is not selected, an edge gateway firewall rule must exist to allow access to the L7 VIP for those protocols. When Acceleration Enabled is selected and the server pool is in non-transparent mode, an SNAT rule is added, so you must ensure that the firewall is enabled on the edge gateway.
- `lb_logging_enabled` - (Optional) Enables the edge gateway load balancer to collect traffic logs. Default is `false`.
- `lb_loglevel` - (Optional) Choose the severity of events to be logged. One of `emergency`, `alert`, `critical`, `error`, `warning`, `notice`, `info`, `debug`
- `fw_enabled` (Optional) Enable firewall. Default `true`. **Note:** Disabling

Firewall will also disable NAT and other NAT dependent features like Load Balancer.
- `fw_default_rule_logging_enabled` (Optional) Enable default firewall rule (last in the processing order) logging. Default `false`.
- `fw_default_rule_action` (Optional) Default firewall rule (last in the processing order) action. One of `accept` or `deny`. Default `deny`.

» External Network

- `name` (Required) - Name of existing external network
- `enable_rate_limit` (Optional) - `True` if rate limiting should be applied on this interface. Default is `false`.
- `incoming_rate_limit` (Optional) - Incoming rate limit in Mbps.
- `outgoing_rate_limit` (Optional) - Outgoing rate limit in Mbps.
- `subnet` (Required) - One or more blocks of External Network Subnet.

Note: Rate limiting works only with external networks backed by distributed portgroups.

» External Network Subnet

- `gateway` (Required) - Gateway for a subnet in external network
- `netmask` (Required) - Netmask of a subnet in external network
- `ip_address` (Optional) - IP address to assign to edge gateway interface (will be auto-assigned if unspecified)
- `use_for_default_route` (Optional) - Should this network be used as default gateway on edge gateway. Default is `false`.
- `suballocate_pool` (Optional) - One or more blocks of ip ranges in the subnet to be sub-allocated

» External Network Subnet Sub-Allocation

- `start_address` (Required) - Start IP address of a range
- `end_address` (Required) - End IP address of a range

» Attribute Reference

The following attributes are exported on this resource:

- `default_external_network_ip` (*v2.6+*) - IP address of edge gateway used for default network
- `external_network_ips` (*v2.6+*) - A list of IP addresses assigned to edge gateway interfaces connected to external networks.

» Importing

Supported in provider *v2.5+*

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing edge gateway can be imported into this resource via supplying its path. The path for this resource is made of org-name.vdc-name.edge-name For example, using this structure, representing an edge gateway that was **not** created using Terraform:

```
resource "vcd_edgegateway" "tf-edgegateway" {
  name              = "my-edge-gw"
  org               = "my-org"
  vdc               = "my-vdc"
  configuration     = "COMPUTE"
  external_networks = ["COMPUTE"]
}
```

You can import such edge gateway into terraform state using this command

```
terraform import vcd_edgegateway.tf-edgegateway my-org.my-vdc.my-edge-gw
```

// Note: the separator can be changed using Provider.import_separator or variable VCD_IMPORT_SEPARATOR

After importing, if you run `terraform plan` you will see the rest of the values and modify the script accordingly for further operations.

» vcd_edgegateway_vpn

Provides a vCloud Director IPsec VPN. This can be used to create, modify, and delete VPN settings and rules.

» Example Usage

```
resource "vcd_edgegateway_vpn" "vpn" {
  edge_gateway        = "Internet_01(nti0000bi2_123-456-2)"
  name                = "west-to-east"
  description         = "Description"
  encryption_protocol = "AES256"
  mtu                 = 1400
  peer_id             = "64.121.123.11"
  peer_ip_address     = "64.121.123.11"
  local_id            = "64.121.123.10"
  local_ip_address    = "64.121.123.10"
  shared_secret       = "***********************"

  peer_subnets {
    peer_subnet_name    = "DMZ_WEST"
    peer_subnet_gateway = "10.0.10.1"
    peer_subnet_mask    = "255.255.255.0"
  }

  peer_subnets {
    peer_subnet_name    = "WEB_WEST"
    peer_subnet_gateway = "10.0.20.1"
    peer_subnet_mask    = "255.255.255.0"
  }

  local_subnets {
    local_subnet_name    = "DMZ_EAST"
    local_subnet_gateway = "10.0.1.1"
    local_subnet_mask    = "255.255.255.0"
  }

  local_subnets {
    local_subnet_name    = "WEB_EAST"
    local_subnet_gateway = "10.0.22.1"
    local_subnet_mask    = "255.255.255.0"
  }
}
```

» Argument Reference

The following arguments are supported:

- `edge_gateway` - (Required) The name of the edge gateway on which to apply the Firewall Rules
- `name` - (Required) The name of the VPN
- `description` - (Required) A description for the VPN
- `encryption_protocol` - (Required) - E.g. `AES256`
- `local_ip_address` - (Required) - Local IP Address
- `local_id` - (Required) - Local ID
- `mtu` - (Required) - The MTU setting
- `peer_ip_address` - (Required) - Peer IP Address
- `peer_id` - (Required) - Peer ID
- `shared_secret` - (Required) - Shared Secret
- `local_subnets` - (Required) - List of Local Subnets see Local Subnets below for details.
- `peer_subnets` - (Required) - List of Peer Subnets see Peer Subnets below

for details.

- `org` - (Optional; *v2.0+*) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional; *v2.0+*) The name of VDC to use, optional if defined at provider level

» Local Subnets

Each Local Subnet supports the following attributes:

- `local_subnet_name` - (Required) Name of the local subnet
- `local_subnet_gateway` - (Required) Gateway of the local subnet
- `local_subnet_mask` - (Required) Subnet mask of the local subnet

» Peer Subnets

Each Peer Subnet supports the following attributes:

- `peer_subnet_name` - (Required) Name of the peer subnet
- `peer_subnet_gateway` - (Required) Gateway of the peer subnet
- `peer_subnet_mask` - (Required) Subnet mask of the peer subnet

» vcd_vapp

Provides a vCloud Director vApp resource. This can be used to create, modify, and delete vApps.

» Example Usage

Example with more than one VM under a vApp.

```
resource "vcd_network_direct" "net" {
  name             = "net"
  external_network = "corp-network"
}

resource "vcd_vapp" "web" {
  name = "web"

  metadata = {
    CostAccount = "Marketing Department"
  }
  depends_on = ["vcd_network_direct.net"]
}

resource "vcd_vapp_vm" "web1" {
  vapp_name     = "${vcd_vapp.web.name}"
  name          = "web1"
  catalog_name  = "Boxes"
  template_name = "lampstack-1.10.1-ubuntu-10.04"
  memory        = 2048
  cpus          = 1

  network_name = "net"
  ip           = "10.10.104.161"

  guest_properties = {
    "vapp.property1"   = "value1"
    "vapp.property2"   = "value2"
  }

  depends_on = ["vcd_vapp.web"]
}

resource "vcd_vapp_vm" "web2" {
  vapp_name     = "${vcd_vapp.web.name}"
  name          = "web2"
  catalog_name  = "Boxes"
  template_name = "lampstack-1.10.1-ubuntu-10.04"
  memory        = 2048
  cpus          = 1

  network_name = "net"
  ip           = "10.10.104.162"

  depends_on = ["vcd_vapp.web"]
}
```

» Example of vApp with single VM

Not recommended in v2.0+ : in the earlier version of the provider it was possible to define a vApp with a single VM in one resource, but it is not recommended as of *v2.0+* provider. Please define vApp and VM in separate resources instead. The implicit inclusion of one VM in a vApp is **Deprecated in 2.5**

```
resource "vcd_network_routed" "net" {
  # ...
}

resource "vcd_vapp" "web" {
  name          = "web"
  catalog_name  = "Boxes"
  template_name = "lampstack-1.10.1-ubuntu-10.04"
  memory        = 2048
  cpus          = 1

  network_name = "${vcd_network.net.name}"
  ip           = "10.10.104.160"

  metadata = {
    role    = "web"
    env     = "staging"
    version = "v1"
  }

  ovf {
    hostname = "web"
  }

  depends_on = ["vcd_network_routed.net"]
}
```

» Example of Empty vApp with no VMs

```
resource "vcd_network_routed" "net" {
  # ...
}

resource "vcd_vapp" "web" {
  name = "web"

  metadata = {
    boss = "Why is this vApp empty?"
    john = "I don't really know. Maybe somebody did forget to clean it up."
  }

  depends_on = ["vcd_network_routed.net"]
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) A unique name for the vApp
- `org` - (Optional; *v2.0+*) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional; *v2.0+*) The name of VDC to use, optional if defined at provider level
- `power_on` - (Optional) A boolean value stating if this vApp should be powered on. Default is `true`
- `metadata` - (Optional) Key value map of metadata to assign to this vApp. Key and value can be any string. (Since *v2.2+* metadata is added directly to vApp instead of first VM in vApp)
- `guest_properties` - (Optional; *v2.5+*) Key value map of vApp guest properties
- `href` - (Computed) The vApp Hyper Reference
- `status` - (Computed; *v2.5+*) The vApp status as a numeric code
- `status_text` - (Computed; *v2.5+*) The vApp status as text.

» Deprecated arguments

The following arguments are deprecated because they refer to the ability of deploying an implicit VM within the vApp. The recommended method is now to use the attributes above to set an empty vApp and then use the resource `vcd_vapp_vm` to deploy one or more VMs within the vApp.

- `catalog_name` - (Optional; **Deprecated**) The catalog name in which to find the given vApp Template
- `template_name` - (Optional; **Deprecated**) The name of the vApp Template to use
- `memory` - (Optional; **Deprecated**) The amount of RAM (in MB) to allocate to the vApp
- `storage_profile` - (Optional; **Deprecated**) Storage profile to override the default one.
- `cpus` - (Optional; **Deprecated**) The number of virtual CPUs to allocate to the vApp
- `initscript` (Optional; **Deprecated**) A script to be run only on initial boot
- `network_name` - (Optional; **Deprecated**) Name of the network this vApp should join. Use the `network` block in `vcd_vapp_vm` instead.

- `ip` - (Optional; **Deprecated**) The IP to assign to this vApp. Must be an IP address or one of dhcp, allocated or none. If given the address must be within the `static_ip_pool` set for the network. If left blank, and the network has `dhcp_pool` set with at least one available IP then this will be set with DHCP. Use the `network` block in `vcd_vapp_vm` instead.
- `ovf` - (Optional; **Deprecated**) Key value map of ovf parameters to assign to VM product section. Use `guest_properties` either in this resource or in `vcd_vapp_vm` instead. **Note** `ovf` attribute sets guest properties on the first VM using a legacy ability of this resource to spawn 1 VM.
- `accept_all_eulas` - (Optional; *v2.0+*; **Deprecated**) Automatically accept EULA if OVA has it. Default is `true`

» Importing

Supported in provider *v2.5+*

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing vApp can be imported into this resource via supplying its path. The path for this resource is made of org-name.vdc-name.vapp-name For example, using this structure, representing a vApp that was **not** created using Terraform:

```
resource "vcd_vapp" "tf-vapp" {
  name              = "my-vapp"
  org               = "my-org"
  vdc               = "my-vdc"
}
```

You can import such vapp into terraform state using this command

```
terraform import vcd_vapp.tf-vapp my-org.my-vdc.my-vapp
```

NOTE: the default separator (.) can be changed using Provider.import_separator or variable VCD_IMPORT_SEPARATOR

After importing, if you run `terraform plan` you will see the rest of the values and modify the script accordingly for further operations.

» vcd_vapp_network

Provides a vCloud Director vApp isolated Network. This can be used to create and delete internal networks for vApps to connect. This network is not attached to external networks or routers.

Supported in provider *v2.1+*

» Example Usage

```
resource "vcd_vapp_network" "vappNet" {
  org = "my-org" #Optional
  vdc = "my-vdc" #Optional

  name               = "my-net"
  vapp_name          = "my-vapp"
  gateway            = "192.168.2.1"
  netmask            = "255.255.255.0"
  dns1               = "192.168.2.1"
  dns2               = "192.168.2.2"
  dns_suffix         = "mybiz.biz"
  guest_vlan_allowed = true

  static_ip_pool {
    start_address = "192.168.2.51"
    end_address   = "192.168.2.100"
  }

  dhcp_pool {
    start_address = "192.168.2.2"
    end_address   = "192.168.2.50"
  }
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional; *v2.0+*) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations.
- `vdc` - (Optional; *v2.0+*) The name of VDC to use, optional if defined at provider level.
- `name` - (Required) A unique name for the network.
- `vapp_name` - (Required) The vApp this VM should belong to.
- `netmask` - (Optional) The netmask for the new network. Default is `255.255.255.0`.
- `gateway` (Optional) The gateway for this network.
- `dns1` - (Optional) First DNS server to use. Default is `8.8.8.8`.
- `dns2` - (Optional) Second DNS server to use. Default is `8.8.4.4`.
- `dns_suffix` - (Optional) A FQDN for the virtual machines on this network.

- `guest_vlan_allowed` (Optional) True if Network allows guest VLAN tagging. This value supported from vCD version 9.0
- `static_ip_pool` - (Optional) A range of IPs permitted to be used as static IPs for virtual machines; see IP Pools below for details.
- `dhcp_pool` - (Optional) A range of IPs to issue to virtual machines that don't have a static IP; see IP Pools below for details.

» IP Pools

Static IP Pools and DHCP Pools support the following attributes:

- `start_address` - (Required) The first address in the IP Range.
- `end_address` - (Required) The final address in the IP Range.

DHCP Pools additionally support the following attributes:

- `default_lease_time` - (Optional) The default DHCP lease time to use. Defaults to `3600`.
- `max_lease_time` - (Optional) The maximum DHCP lease time to use. Defaults to `7200`.
- `enabled` - (Optional) Allows to enable or disable service. Default is true.

» vcd_vapp_vm

Provides a vCloud Director VM resource. This can be used to create, modify, and delete VMs within a vApp.

Note: To make sure resources are created in the right order and both plan apply and destroy succeeds, use the `depends_on` clause (see example below)

» Example Usage

```
resource "vcd_network_direct" "net" {
  name             = "net"
  external_network = "corp-network"
}

resource "vcd_vapp" "web" {
  name = "web"

  depends_on = ["vcd_network_direct.net"]
}

resource "vcd_vapp_vm" "web1" {
  vapp_name     = "${vcd_vapp.web.name}"
  name          = "web1"
  catalog_name  = "Boxes"
  template_name = "lampstack-1.10.1-ubuntu-10.04"
  memory        = 2048
  cpus          = 2
  cpu_cores     = 1

  metadata = {
    role    = "web"
    env     = "staging"
    version = "v1"
    my_key  = "my value"
  }

  guest_properties = {
    "guest.hostname"   = "my-host"
    "another.var.name" = "var-value"
  }

  network {
    type               = "org"
    name               = "net"
    ip                 = "10.10.104.161"
    ip_allocation_mode = "MANUAL"
    is_primary         = true
  }

  depends_on = ["vcd_vapp.web"]
}

resource "vcd_vapp_vm" "web2" {
  vapp_name     = "${vcd_vapp.web.name}"
  name          = "web2"
  catalog_name  = "Boxes"
  template_name = "lampstack-1.10.1-ubuntu-10.04"
  memory        = 2048
  cpus          = 1

  metadata = {
    role    = "web"
    env     = "staging"
    version = "v1"
    my_key  = "my value"
  }
  network {
    type               = "org"
    name               = "net"
    ip                 = "10.10.104.162"
    ip_allocation_mode = "MANUAL"
    is_primary         = true
  }

  network {
    type               = "vapp"
    name               = "vapp-network"
    ip_allocation_mode = "POOL"
  }

  network {
    type               = "none"
    ip_allocation_mode = "NONE"
  }

  disk {
    name        = "logDisk1"
    bus_number  = 1
    unit_number = 0
  }

  disk {
    name        = "logDisk2"
    bus_number  = 1
    unit_number = 1
  }

  guest_properties = {
    "guest.hostname" = "my-hostname"
    "guest.other"    = "another-setting"
  }

  depends_on = ["vcd_vapp.web"]
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional; *v2.0+*) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations

- `vdc` - (Optional; *v2.0+*) The name of VDC to use, optional if defined at provider level
- `vapp_name` - (Required) The vApp this VM belongs to.
- `name` - (Required) A name for the VM, unique within the vApp
- `computer_name` - (Optional; *v2.5+*) Computer name to assign to this virtual machine.
- `catalog_name` - (Required) The catalog name in which to find the given vApp Template
- `template_name` - (Required) The name of the vApp Template to use
- `memory` - (Optional) The amount of RAM (in MB) to allocate to the VM
- `cpus` - (Optional) The number of virtual CPUs to allocate to the VM. Socket count is a result of: virtual logical processors/cores per socket. The default is 1
- `cpu_cores` - (Optional; *v2.1+*) The number of cores per socket. The default is 1
- `metadata` - (Optional; *v2.2+*) Key value map of metadata to assign to this VM
- `initscript` (Optional) Script to run on initial boot or with customization.force=true set
- `storage_profile` (Optional; *v2.6+*) Storage profile to override the default one
- `network_name` - (Optional; **Deprecated** by `network`) Name of the network this VM should connect to.
- `vapp_network_name` - (Optional; v2.1+; **Deprecated** by `network`) Name of the vApp network this VM should connect to.
- `ip` - (Optional; **Deprecated** by `network`) The IP to assign to this vApp. Must be an IP address or one of `dhcp`, `allocated`, or `none`. If given the address must be within the `static_ip_pool` set for the network. If left blank, and the network has `dhcp_pool` set with at least one available IP then this will be set with DHCP.
- `power_on` - (Optional) A boolean value stating if this VM should be powered on. Default is `true`
- `accept_all_eulas` - (Optional; *v2.0+*) Automatically accept EULA if OVA has it. Default is `true`
- `disk` - (Optional; *v2.1+*) Independent disk attachment configuration. See Disk below for details.
- `expose_hardware_virtualization` - (Optional; *v2.2+*) Boolean for exposing full CPU virtualization to the guest operating system so that applications that require hardware virtualization can run on virtual machines without binary translation or paravirtualization. Useful for hypervisor nesting provided underlying hardware supports it. Default is `false`.
- `network` - (Optional; *v2.2+*) A block to define network interface. Multiple can be used. See Network and example for usage details. **Deprecates**: `network_name`, `ip`, `vapp_network_name`.
- `customization` - (Optional; *v2.5+*) A block to define for guest customiza-

tion options. See Customization

- `guest_properties` - (Optional; *v2.5+*) Key value map of guest properties
- `description` - (Computed; *v2.6+*) The VM description. Note: description is read only. Currently, this field has the description of the OVA used to create the VM

» Disk

- `name` - (Required) Independent disk name
- `bus_number` - (Required) Bus number on which to place the disk controller
- `unit_number` - (Required) Unit number (slot) on the bus specified by BusNumber.

» Network

- `type` (Required) Network type, one of: `none`, `vapp` or `org`. `none` creates a NIC with no network attached, `vapp` attaches a vApp network, while `org` attaches organization VDC network.

- `name` (Optional) Name of the network this VM should connect to. Always required except for `type` `NONE`.

- `is_primary` (Optional) Set to true if network interface should be primary. First network card in the list will be primary by default.

- `mac` - (Computed) Mac address of network interface.

- `ip_allocation_mode` (Required) IP address allocation mode. One of `POOL`, `DHCP`, `MANUAL`, `NONE`:

  - `POOL` - Static IP address is allocated automatically from defined static pool in network.
  - `DHCP` - IP address is obtained from a DHCP service. Field `ip` is not guaranteed to be populated. Because of this it may appear after multiple `terraform refresh` operations.
  - `MANUAL` - IP address is assigned manually in the `ip` field. Must be valid IP address from static pool.
  - `NONE` - No IP address will be set because VM will have a NIC without network.

- `ip` (Optional, Computed) Settings depend on `ip_allocation_mode`. Field requirements for each `ip_allocation_mode` are listed below:

  - `ip_allocation_mode=POOL` - `ip` value must be omitted or empty string "". Empty string may be useful when doing HCL variable interpolation. Field `ip` will be populated with an assigned IP from static pool after run.

  - `ip_allocation_mode=DHCP` - `ip` value must be omitted or empty string "". Field `ip` is not guaranteed to be populated after run due to the VM lacking VMware tools or not working properly with DHCP. Because of this `ip` may also appear after multiple `terraform refresh` operations when is reported back to vCD.
  - `ip_allocation_mode=MANUAL` - `ip` value must be valid IP address from a subnet defined in `static pool` for network.
  - `ip_allocation_mode=NONE` - `ip` field can be omitted or set to an empty string "". Empty string may be useful when doing HCL variable interpolation.

» Customization

- `force` (Optional) **Warning.** `true` value will cause the VM to reboot on every `apply` operation. This field works as a flag and triggers force customization when `true` during an update (`terraform apply`) every time. It never complains about a change in statefile. Can be used when guest customization is needed after VM configuration (e.g. NIC change, customization options change, etc.) and then set back to `false`. **Note.** It will not have effect when `power_on` field is set to `false`. See example workflow below.

» Example forced customization workflow

Step 1 - Setup VM:

```
resource "vcd_vapp_vm" "web2" {
  vapp_name     = "${vcd_vapp.web.name}"
  name          = "web2"
  catalog_name  = "Boxes"
  template_name = "lampstack-1.10.1-ubuntu-10.04"
  memory        = 2048
  cpus          = 1

  network {
    type               = "org"
    name               = "net"
    ip                 = "10.10.104.162"
    ip_allocation_mode = "MANUAL"
  }
}
```

Step 2 - Change VM configuration and force customization (VM will be rebooted during `terraform apply`):

```
resource "vcd_vapp_vm" "web2" {
//...
  network {
    type               = "org"
    name               = "net"
    ip_allocation_mode = "DHCP"
  }

  customization {
    force = true
  }
}
```

Step 3 - Once customization is done, set the force customization flag to false (or remove it) to prevent forcing customization on every `terraform apply` command:

```
resource "vcd_vapp_vm" "web2" {
//...
  network {
    type               = "org"
    name               = "net"
    ip_allocation_mode = "DHCP"
  }

  customization {
    force = false
  }
}
```

» Importing

Supported in provider *v2.6+*

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing VM can be imported into this resource via supplying its path. The path for this resource is made of org-name.vdc-name.vapp-name.vm-name For example, using this structure, representing a VM that was **not** created using Terraform:

```
resource "vcd_vapp_vm" "tf-vm" {
  name              = "my-vm"
  org               = "my-org"
  vdc               = "my-vdc"
  vapp_name         = "my-vapp"
}
```

You can import such vapp into terraform state using this command

```
terraform import vcd_vapp_vm.tf-vm my-org.my-vdc.my-vapp.my-vm
```

NOTE: the default separator (.) can be changed using Provider.import_separator or variable VCD_IMPORT_SEPARATOR

After importing, the data for this VM will be in the state file (`terraform.tfstate`). If you want to use this resource for further operations, you will need to integrate it with data from the state file, and with some data that is used to create the VM, such as `catalog_name`, `template_name`.

» vcd_org_vdc

Provides a vCloud Director Organization VDC resource. This can be used to create and delete an Organization VDC. Requires system administrator privileges.

Supported in provider *v2.2+*

» Example Usage

```
provider "vcd" {
  user     = "${var.admin_user}"
  password = "${var.admin_password}"
  org      = "System"
  url      = "https://AcmeVcd/api"
}

resource "vcd_org_vdc" "my-vdc" {
  name        = "my-vdc"
  description = "The pride of my work"
  org         = "my-org"

  allocation_model = "ReservationPool"
  network_pool_name = "vDC1-VXLAN-NP"
  provider_vdc_name = "vDC1"

  compute_capacity {
    cpu {
      allocated = 2048
    }

    memory {
      allocated = 2048
    }
  }

  storage_profile {
    name     = "storage-name"
    limit    = 10240
    default  = true
  }

  metadata = {
    role    = "customerName"
    env     = "staging"
    version = "v1"
  }

  enabled                  = true
  enable_thin_provisioning = true
  enable_fast_provisioning = true
  delete_force             = true
  delete_recursive         = true
}
```

» Argument Reference

The following arguments are supported:

Note: Only part of fields are read if user is Organization administrator. With System Admin user all fields are populated.

- `org` - (Optional) Organization to create the VDC in, optional if defined at provider level
- `name` - (Required) VDC name
- `description` - (Optional) VDC friendly description
- `provider_vdc_name` - (Required, System Admin) Name of the Provider VDC from which this organization VDC is provisioned.
- `allocation_model` - (Required) The allocation model used by this VDC; must be one of {AllocationVApp (”Pay as you go”), AllocationPool (”Allocation pool”), ReservationPool (”Reservation pool”)}
- `compute_capacity` - (Required) The compute capacity allocated to this VDC. See Compute Capacity below for details.
- `nic_quota` - (Optional) Maximum number of virtual NICs allowed in this VDC. Defaults to 0, which specifies an unlimited number.
- `network_quota` - (Optional) Maximum number of network objects that can be deployed in this VDC. Defaults to 0, which means no networks can be deployed.

- `vm_quota` - (Optional) The maximum number of VMs that can be created in this VDC. Includes deployed and undeployed VMs in vApps and vApp templates. Defaults to 0, which specifies an unlimited number.
- `enabled` - (Optional) True if this VDC is enabled for use by the organization VDCs. Default is true.
- `storage_profile` - (Required, System Admin) Storage profiles supported by this VDC. See Storage Profile below for details.
- `memory_guaranteed` - (Optional, System Admin) Percentage of allocated memory resources guaranteed to vApps deployed in this VDC. For example, if this value is 0.75, then 75% of allocated resources are guaranteed. Required when AllocationModel is AllocationVApp or AllocationPool. When Allocation model is AllocationPool minimum value is 0.2. If left empty, vCD sets a value.
- `cpu_guaranteed` - (Optional, System Admin) Percentage of allocated CPU resources guaranteed to vApps deployed in this VDC. For example, if this value is 0.75, then 75% of allocated resources are guaranteed. Required when AllocationModel is AllocationVApp or AllocationPool. If left empty, vCD sets a value.
- `cpu_speed` - (Optional, System Admin) Specifies the clock frequency, in Megahertz, for any virtual CPU that is allocated to a VM. A VM with 2 vCPUs will consume twice as much of this value. Ignored for ReservationPool. Required when AllocationModel is AllocationVApp or AllocationPool, and may not be less than 256 MHz. Defaults to 1000 MHz if value isn't provided.
- `metadata` - (Optional; *v2.4+*) Key value map of metadata to assign to this VDC
- `enable_thin_provisioning` - (Optional, System Admin) Boolean to request thin provisioning. Request will be honored only if the underlying data store supports it. Thin provisioning saves storage space by committing it on demand. This allows over-allocation of storage.
- `enable_fast_provisioning` - (Optional, System Admin) Request fast provisioning. Request will be honored only if the underlying datastore supports it. Fast provisioning can reduce the time it takes to create virtual machines by using vSphere linked clones. If you disable fast provisioning, all provisioning operations will result in full clones.
- `network_pool_name` - (Optional, System Admin) Reference to a network pool in the Provider VDC. Required if this VDC will contain routed or isolated networks.
- `allow_over_commit` - (Optional) Set to false to disallow creation of the VDC if the AllocationModel is AllocationPool or ReservationPool and the ComputeCapacity you specified is greater than what the backing Provider VDC can supply. Default is true.
- `enable_vm_discovery` - (Optional) If true, discovery of vCenter VMs is enabled for resource pools backing this VDC. If false, discovery is disabled. If left unspecified, the actual behaviour depends on enablement at the organization level and at the system level.

- `delete_force` - (Required) When destroying use `delete_force=True` to remove a VDC and any objects it contains, regardless of their state.
- `delete_recursive` - (Required) When destroying use `delete_recursive=True` to remove the VDC and any objects it contains that are in a state that normally allows removal.

» Storage Profile

- `name` - (Required) Name of Provider VDC storage profile.
- `enabled` - (Optional) True if this storage profile is enabled for use in the VDC. Default is true.
- `limit` - (Required) Maximum number of MB allocated for this storage profile. A value of 0 specifies unlimited MB.
- `default` - (Required) True if this is default storage profile for this VDC. The default storage profile is used when an object that can specify a storage profile is created with no storage profile specified.

» Compute Capacity

Capacity must be specified twice, once for `memory` and another for `cpu`. Each has the same structure:

- `allocated` - (Optional) Capacity that is committed to be available. Value in MB or MHz. Used with AllocationPool (”Allocation pool”) and ReservationPool (”Reservation pool”).
- `limit` - (Optional) Capacity limit relative to the value specified for Allocation. It must not be less than that value. If it is greater than that value, it implies over provisioning. A value of 0 specifies unlimited units. Value in MB or MHz. Used with AllocationVApp (”Pay as you go”).

» Importing

Supported in provider *v2.5+*

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing an organization VDC can be imported into this resource via supplying the full dot separated path to VDC. An example is below:

```
terraform import vcd_org_vdc.my-vdc my-org.my-vdc
```

NOTE: the default separator (.) can be changed using Provider.import_separator or variable VCD_IMPORT_SEPARATOR

After that, you can expand the configuration file and either update or delete the VDC as needed. Running `terraform plan` at this stage will show the difference between the minimal configuration file and the VDC's stored properties.

» vcd_lb_service_monitor

Provides a vCloud Director Edge Gateway Load Balancer Service Monitor resource. A service monitor defines health check parameters for a particular type of network traffic. It can be associated with a pool. Pool members are monitored according to the service monitor parameters.

Note: To make load balancing work one must ensure that load balancing is enabled on edge gateway (edge gateway must be advanced). This depends on NSX version to work properly. Please refer to VMware Product Interoperability Matrices to check supported vCloud director and NSX for vSphere configurations.

Note: The vCloud Director API for NSX supports a subset of the operations and objects defined in the NSX vSphere API Guide. The API supports NSX 6.2, 6.3, and 6.4.

Supported in provider *v2.4+*

» Example Usage

```
resource "vcd_lb_service_monitor" "monitor" {
  org          = "my-org"
  vdc          = "my-org-vdc"
  edge_gateway = "my-edge-gw"

  name        = "http-monitor"
  interval    = "5"
  timeout     = "20"
  max_retries = "3"
  type        = "http"
  method      = "GET"
  url         = "/health"
  send        = "{\"key\": \"value\"}"
  extension = {
    content-type = "application/json"
    linespan     = ""
  }
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `edge_gateway` - (Required) The name of the edge gateway on which the service monitor is to be created
- `name` - (Required) Service Monitor name
- `interval` - (Optional) Interval in seconds at which a server is to be monitored using the specified Method. Defaults to 10
- `timeout` - (Optional) Maximum time in seconds within which a response from the server must be received. Defaults to 15
- `max_retries` - (Optional) Number of times the specified monitoring Method must fail sequentially before the server is declared down. Defaults to 3
- `type` - (Required) Select the way in which you want to send the health check request to the server — `http`, `https`, `tcp`, `icmp`, or `udp`. Depending on the type selected, the remaining attributes are allowed or not
- `method` - (Optional) For types `http` and `https`. Select http method to be used to detect server status. One of OPTIONS, GET, HEAD, POST, PUT, DELETE, TRACE, or CONNECT
- `url` - (Optional) For types `http` and `https`. URL to be used in the server status request
- `send` - (Optional) For types `http`, `https`, and `udp`. The data to be sent.
- `expected` - (Optional) For types `http` and `https`. String that the monitor expects to match in the status line of the HTTP or HTTPS response (for example, `HTTP/1.1`)
- `receive` - (Optional) For types `http`, `https`, and `udp`. The string to be matched in the response content. **Note**: When `expected` is not matched, the monitor does not try to match the Receive content
- `extension` - (Optional) A map of advanced monitor parameters as key=value pairs (i.e. `max-age=SECONDS`, `invert-regex`) **Note**: When you need a value of `key` only format just set value to empty string (i.e. `linespan = ""`)

» Attribute Reference

The following attributes are exported on the base level of this resource:

- `id` - The NSX ID of the load balancer service monitor

» Importing

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing load balancer service monitor can be imported into this resource via supplying the full dot separated path for load balancer service monitor. An example is below:

```
terraform import vcd_lb_service_monitor.imported my-org.my-org-vdc.my-edge-gw.my-lb-service-
```

The above would import the service monitor named `my-lb-service-monitor` that is defined on edge gateway `my-edge-gw` which is configured in organization named `my-org` and vDC named `my-org-vdc`.

» vcd_lb_server_pool

Provides a vCloud Director Edge Gateway Load Balancer Server Pool resource. A Server Pool can have a group of backend servers set (defined as pool members), manages load balancer distribution methods, and may have a service monitor attached to it for health check parameters.

Note: To make load balancing work one must ensure that load balancing is enabled on edge gateway. This depends on NSX version to work properly. Please refer to VMware Product Interoperability Matrices to check supported vCloud director and NSX for vSphere configurations.

Note: The vCloud Director API for NSX supports a subset of the operations and objects defined in the NSX vSphere API Guide. The API supports NSX 6.2, 6.3, and 6.4.

Supported in provider *v2.4+*

» Example Usage 1 (Simple Server Pool without Service Monitor)

```
resource "vcd_lb_server_pool" "web-servers" {
  org          = "my-org"
  vdc          = "my-org-vdc"
  edge_gateway = "my-edge-gw"

  name      = "web-servers"
  algorithm = "round-robin"

  member {
    condition       = "enabled"
    name            = "member1"
    ip_address      = "1.1.1.1"
    port            = 8443
    monitor_port    = 9000
    weight          = 1
    min_connections = 0
    max_connections = 100
  }
}
```

» Example Usage 2 (Server Pool with multiple members, algorithm parameters, and existing Service Monitor as data source)

```
data "vcd_lb_service_monitor" "web-monitor" {
  org          = "my-org"
  vdc          = "my-org-vdc"
  edge_gateway = "my-edge-gw"

  name = "existing-web-monitor-name"
}

resource "vcd_lb_server_pool" "web-servers" {
  org          = "my-org"
  vdc          = "my-org-vdc"
  edge_gateway = "my-edge-gw"

  name                 = "web-servers"
  description          = "description"
  algorithm            = "httpheader"
  algorithm_parameters = "headerName=host"
  enable_transparency  = "true"

  monitor_id = "${data.vcd_lb_service_monitor.web-monitor.id}"

  member {
    condition       = "enabled"
    name            = "member1"
    ip_address      = "1.1.1.1"
    port            = 8443
    monitor_port    = 9000
    weight          = 1
    min_connections = 0
    max_connections = 100
  }

  member {
    condition       = "drain"
    name            = "member2"
    ip_address      = "2.2.2.2"
    port            = 7000
    monitor_port    = 4000
    weight          = 2
    min_connections = 6
    max_connections = 8
  }
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `edge_gateway` - (Required) The name of the edge gateway on which the server pool is to be created
- `name` - (Required) Server Pool name
- `description` - (Optional) Server Pool description
- `algorithm` - (Required) Server Pool load balancing method. Can be one of `ip-hash`, `round-robin`, `uri`, `leastconn`, `url`, or `httpheader`
- `algorithm_parameters -` (Optional) Valid only when `algorithm` is `httpheader` or `url`. The `httpheader` algorithm parameter has one option `headerName=<name>` while the `url` algorithm parameter has option `urlParam=<url>`.
- `enable_transparency` - (Optional) When transparency is `false` (default) backend servers see the IP address of the traffic source as the internal IP address of the load balancer. When it is `true` the source IP address is the actual IP address of the client and the edge gateway must be set as the default gateway to ensure that return packets go through the edge gateway.
- `monitor_id` - (Optional) `vcd_lb_service_monitor` resource `id` to attach to server pool for health check parameters
- `member` - (Optional) A block to define server pool members. Multiple can be used. See Member and example for usage details.

» Member

- condition - (Required) State of member in a pool. One of enabled, disabled, or drain. When member condition is set to drain it stops taking new connections and calls, while it allows its sessions on existing connections to continue until they naturally end. This allows to gracefully remove member node from load balancing rotation.
- name - (Required) Member name
- ip_address - (Required) Member IP address
- port - (Required) The port at which the member is to receive traffic from the load balancer.
- monitor_port - (Required) Monitor Port at which the member is to receive health monitor requests. **Note:** can be the same as port
- weight - (Required) The proportion of traffic this member is to handle. Must be an integer in the range 1-256.
- min_connections - (Optional) The maximum number of concurrent connections the member can handle. **Note:** when the number of incoming requests exceeds the maximum, requests are queued and the load balancer waits for a connection to be released.
- max_connections - (Optional) The minimum number of concurrent connections a member must always accept.

» Attribute Reference

The following attributes are exported on this resource:

- id - The NSX ID of the load balancer server pool

Additionally each of members defined in blocks expose their own id fields as well

» Importing

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing load balancer server pool can be imported into this resource via supplying the full dot separated path for load balancer service monitor. An example is below:

```
terraform import vcd_lb_server_pool.imported my-org.my-org-vdc.my-edge-gw.my-lb-server-pool
```

The above would import the server pool named my-lb-server-pool that is defined on edge gateway my-edge-gw which is configured in organization named my-org and vDC named my-org-vdc.

» vcd_lb_app_profile

Provides a vCloud Director Edge Gateway Load Balancer Application Profile resource. An application profile defines the behavior of the load balancer for a particular type of network traffic. After configuring a profile, you associate it with a virtual server. The virtual server then processes traffic according to the values specified in the profile.

Note: This resource does not currently support attaching Pool and Virtual Server certificates. The `enable_pool_side_ssl` only toggles the option, but does not setup certificates.

Note: To make load balancing work one must ensure that load balancing is enabled on edge gateway (edge gateway must be advanced). This depends on NSX version to work properly. Please refer to VMware Product Interoperability Matrices to check supported vCloud director and NSX for vSphere configurations.

Note: The vCloud Director API for NSX supports a subset of the operations and objects defined in the NSX vSphere API Guide. The API supports NSX 6.2, 6.3, and 6.4.

Supported in provider *v2.4+*

» Example Usage 1 (TCP Application Profile)

```
resource "vcd_lb_app_profile" "tcp" {
  org          = "my-org"
  vdc          = "my-org-vdc"
  edge_gateway = "my-edge-gw"

  name = "tcp-app-profile"
  type = "tcp"
}
```

» Example Usage 2 (HTTP Cookie based Application Profile)

```
resource "vcd_lb_app_profile" "http" {
  org          = "my-org"
  vdc          = "my-org-vdc"
  edge_gateway = "my-edge-gw"

  name = "http-profile"
  type = "http"
  http_redirect_url              = "/service-one"
  persistence_mechanism          = "cookie"
  cookie_name                    = "JSESSIONID"
  cookie_mode                    = "insert"
  insert_x_forwarded_http_header = "true"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `edge_gateway` - (Required) The name of the edge gateway on which the application profile is to be created
- `name` - (Required) Application profile name
- `type` - (Required) Protocol type used to send requests to the server. One of `tcp`, `udp`, `http`, or `https`
- `enable_ssl_passthrough` - (Optional) Enable SSL authentication to be passed through to the virtual server. Otherwise SSL authentication takes place at the destination address
- `http_redirect_url` - (Optional) The URL to which traffic that arrives at the destination address should be redirected. Only applies for types `http` and `https`
- `persistence_mechanism` - (Optional) Persistence mechanism for the profile. One of 'cookie', 'ssl-sessionid', 'sourceip'
- `cookie_name` - (Optional) Used to uniquely identify the session the first time a client accesses the site. The load balancer refers to this cookie when connecting subsequent requests in the session, so that they all go to the same virtual server. Only applies for `persistence_mechanism` 'cookie'
- `cookie_mode` - (Optional) The mode by which the cookie should be inserted. One of 'insert', 'prefix', or 'appsession'
- `expiration` - (Optional) Length of time in seconds that persistence stays in effect
- `insert_x_forwarded_http_header` - (Optional) Enables 'X-Forwarded-For' header for identifying the originating IP address of a client connecting to a Web server through the load balancer. Only applies for types `http` and `https`
- `enable_pool_side_ssl` - (Optional) Enable to define the certificate, CAs, or CRLs used to authenticate the load balancer from the server side. **Note:** This resource does not currently support attaching Pool and Vir-

tual Server certificates therefore this toggle only enables it. To make it fully work certificates must be currently attached manually.

» Attribute Reference

The following attributes are exported on this resource:

- `id` - The NSX ID of the load balancer application profile

» Importing

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing load balancer application profile can be imported into this resource via supplying the full dot separated path for load balancer application profile. An example is below:

```
terraform import vcd_lb_app_profile.imported my-org.my-org-vdc.my-edge-gw.my-lb-app-profile
```

The above would import the application profile named `my-lb-app-profile` that is defined on edge gateway `my-edge-gw` which is configured in organization named `my-org` and vDC named `my-org-vdc`.

» vcd_lb_app_rule

Provides a vCloud Director Edge Gateway Load Balancer Application Rule resource. An application rule allows to directly manipulate and manage IP application traffic with load balancer.

Note: To make load balancing work one must ensure that load balancing is enabled on edge gateway (edge gateway must be advanced). This depends on NSX version to work properly. Please refer to VMware Product Interoperability Matrices to check supported vCloud director and NSX for vSphere configurations.

Note: The vCloud Director API for NSX supports a subset of the operations and objects defined in the NSX vSphere API Guide. The API supports NSX 6.2, 6.3, and 6.4.

Supported in provider *v2.4+*

» Example Usage 1 (Application rule with single line script)

```
resource "vcd_lb_app_rule" "example-one" {
  edge_gateway = "my-edge-gw"
  org          = "my-org"
  vdc          = "my-org-vdc"

  name = "script1"
  script = "acl vmware_page url_beg / vmware redirect location https://www.vmware.com/ ifvmw
}
```

» Example Usage 2 (Application rule with multi line script)

```
resource "vcd_lb_app_rule" "example-two" {
  edge_gateway = "my-edge-gw"
  org          = "my-org"
  vdc          = "my-org-vdc"
  name         = "script1"
  script = <<-EOT
    acl vmware_page url_beg / vmware redirect location https://www.vmware.com/ ifvmware_page
    acl other_page2 url_beg / other2 redirect location https://www.other2.com/ ifother_page2
    acl hello payload(0,6) -m bin 48656c6c6f0a
  EOT
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `edge_gateway` - (Required) The name of the edge gateway on which the application rule is to be created
- `name` - (Required) Application rule name
- `script` - (Required) A multiline application rule script. Terraform's HEREDOC syntax may be useful for multiline scripts. **Note:** For information on the application rule syntax, see more in vCloud Director documentation

» Attribute Reference

The following attributes are exported on this resource:

- `id` - The NSX ID of the load balancer application rule

» Importing

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing load balancer application rule can be imported into this resource via supplying the full dot separated path for load balancer application rule. An example is below:

```
terraform import vcd_lb_app_rule.imported my-org.my-org-vdc.my-edge-gw.my-lb-app-rule
```

The above would import the application rule named `my-lb-app-rule` that is defined on edge gateway `my-edge-gw` which is configured in organization named `my-org` and vDC named `my-org-vdc`.

» vcd_lb_virtual_server

Provides a vCloud Director edge gateway load balancer virtual server resource. Adds an edge gateway internal or uplink interface as a virtual server. A virtual server has a public IP address and services all incoming client requests.

Note: To make load balancing work one must ensure that load balancing is enabled on edge gateway (edge gateway must be advanced). This depends on NSX version to work properly. Please refer to VMware Product Interoperability Matrices to check supported vCloud director and NSX for vSphere configurations.

Note: The vCloud Director API for NSX supports a subset of the operations and objects defined in the NSX vSphere API Guide. The API supports NSX 6.2, 6.3, and 6.4.

Supported in provider *v2.4+*

» Example Usage 1 (HTTP virtual server)

```
resource "vcd_lb_virtual_server" "http" {
  org          = "my-org"
  vdc          = "my-org-vdc"
  edge_gateway = "my-edge-gw"

  name       = "http-virtual-server"
  ip_address = "1.1.1.1" # Edge gateway uplink interface IP
  protocol   = "http"    # Must be the same as specified in application profile
  port       = 80

  app_profile_id = "${vcd_lb_app_profile.http.id}"
  server_pool_id = "${vcd_lb_server_pool.web-servers.id}"
  app_rule_ids   = ["${vcd_lb_app_rule.redirect.id}", "${vcd_lb_app_rule.language.id}"]
}
```

» Example Usage 2 (Complete load balancer setup)

```
variable "org" {
  default = "my-org"
}

variable "vdc" {
  default = "my-org-vdc"
}

variable "edge_gateway" {
  default = "my-edge-gw"
}

variable "protocol" {
  default = "http"
}

data "vcd_edgegateway" "mygw" {
  org          = "${var.org}"
  vdc          = "${var.vdc}"
  name         = "${var.edge_gateway.my-edge-gw}"
}

resource "vcd_lb_virtual_server" "http" {
  org          = "${var.org}"
  vdc          = "${var.vdc}"
  edge_gateway = "${var.edge_gateway}"

  name       = "my-virtual-server"
  ip_address = "${data.vcd_edgegateway.mygw.default_external_network_ip}"
  protocol   = "${var.protocol}"
  port       = 8888

  app_profile_id = "${vcd_lb_app_profile.http.id}"
  server_pool_id = "${vcd_lb_server_pool.web-servers.id}"
  app_rule_ids   = ["${vcd_lb_app_rule.redirect.id}"]
}

resource "vcd_lb_service_monitor" "monitor" {
  org          = "${var.org}"
  vdc          = "${var.vdc}"
  edge_gateway = "${var.edge_gateway}"

  name        = "http-monitor"
  interval    = "5"
  timeout     = "20"
  max_retries = "3"
  type        = "${var.protocol}"
  method      = "GET"
  url         = "/health"
  send        = "{\"key\": \"value\"}"
  extension = {
    content-type = "application/json"
    linespan     = ""
  }
}

resource "vcd_lb_server_pool" "web-servers" {
  org          = "${var.org}"
  vdc          = "${var.vdc}"
  edge_gateway = "${var.edge_gateway}"

  name                 = "web-servers"
  description          = "description"
  algorithm            = "httpheader"
  algorithm_parameters = "headerName=host"
  enable_transparency  = "true"

  monitor_id = "${vcd_lb_service_monitor.monitor.id}"

  member {
    condition       = "enabled"
    name            = "member1"
    ip_address      = "1.1.1.1"
    port            = 8443
    monitor_port    = 9000
    weight          = 1
    min_connections = 0
    max_connections = 100
  }

  member {
    condition       = "drain"
    name            = "member2"
    ip_address      = "2.2.2.2"
    port            = 7000
    monitor_port    = 4000
    weight          = 2
    min_connections = 6
    max_connections = 8
  }
}

resource "vcd_lb_app_profile" "http" {
  org          = "${var.org}"
  vdc          = "${var.vdc}"
  edge_gateway = "${var.edge_gateway}"

  name = "http-app-profile"
  type = "${var.protocol}"
}

resource "vcd_lb_app_rule" "redirect" {
  org          = "${var.org}"
  vdc          = "${var.vdc}"
  edge_gateway = "${var.edge_gateway}"

  name   = "redirect"
  script = "acl vmware_page url_beg / vmware redirect location https://www.vmware.com/ ifvmw
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `edge_gateway` - (Required) The name of the edge gateway on which the virtual server is to be created
- `name` - (Required) Virtual server name
- `description` - (Optional) Virtual server description
- `enabled` - (Optional) Defines if the virtual server is enabled. Default `true`
- `enable_acceleration` - (Optional) Defines if the virtual server uses acceleration. Default `false`
- `ip_address` - (Required) Set the IP address that the load balancer listens on
- `protocol` - (Required) Select the protocol that the virtual server accepts. One of `tcp`, `udp`, `http`, or `https` **Note**: You must select the same protocol

used by the selected **Application Profile**

- `port` - (Required) The port number that the load balancer listens on
- `connection_limit` - (Optional) Maximum concurrent connections that the virtual server can process
- `connection_rate_limit` - (Optional) Maximum incoming new connection requests per second
- `server_pool_id` - (Optional) The server pool that the load balancer will use
- `app_profile_id` - (Optional) Application profile ID to be associated with the virtual server
- `app_rule_ids` - (Optional) List of attached application rule IDs

» Attribute Reference

The following attributes are exported on the base level of this resource:

- `id` - The NSX ID of the load balancer virtual server

» Importing

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing load balancer virtual server can be imported into this resource via supplying the full dot separated path for load balancer virtual server. An example is below:

```
terraform import vcd_lb_virtual_server.imported my-org.my-org-vdc.my-edge-gw.my-lb-virtual-s
```

The above would import the virtual server named `my-lb-virtual-server` that is defined on edge gateway `my-edge-gw` which is configured in organization named `my-org` and vDC named `my-org-vdc`.

» vcd_nsxv_dnat

Provides a vCloud Director DNAT resource for advanced edge gateways (NSX-V). This can be used to create, modify, and delete destination NATs to map an external IP/port to an internal IP/port. Replaces `vcd_dnat` resource.

Note: This resource requires advanced edge gateway. For non-advanced edge gateways please use the `vcd_dnat` resource.

Warning: Do not use older `vcd_dnat` resource with this one because it will change IDs and this resource will not be able to lookup rules.

» Example Usage 1 (Minimal input)

```
resource "vcd_nsxv_dnat" "web" {
  org = "my-org" # Optional
  vdc = "my-vdc" # Optional

  edge_gateway = "Edge Gateway Name"
  network_type = "ext"
  network_name = "my-external-network"

  original_address   = "1.1.1.1"
  translated_address = "10.10.10.15"
}
```

» Example Usage 2 (ICMP)

```
resource "vcd_nsxv_dnat" "forIcmp" {
  org = "my-org" # Optional
  vdc = "my-vdc" # Optional

  edge_gateway = "Edge Gateway Name"
  network_name = "my-external-network"
  network_type = "ext"

  original_address   = "78.101.10.20-78.101.10.30"
  translated_address = "10.10.0.5"
  protocol           = "icmp"
  icmp_type          = "router-advertisement"
}
```

» Example Usage 3 (More settings)

```
resource "vcd_nsxv_dnat" "dnat-tcp" {
  org = "my-org" # Optional
  vdc = "my-vdc" # Optional

  edge_gateway = "Edge Gateway Name"
  network_name = "my-external-network"
  network_type = "ext"

  enabled = false
  logging_enabled = true
  description = "My DNAT rule"
  original_address   = "78.101.10.20"
  original_port      = 443

  translated_address = "10.10.0.5"
  translated_port    = 8443
  protocol           = "tcp"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations.
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level.
- `edge_gateway` - (Required) The name of the edge gateway on which to apply the DNAT rule.
- `network_type` - (Required) Type of the network on which to apply the DNAT rule. Possible values `org` or `ext`.
- `network_name` - (Required) The name of the network on which to apply the DNAT rule.
- `enabled` - (Optional) Defines if the rule is enabaled. Default `true`.
- `logging_enabled` - (Optional) Defines if the logging for this rule is enabaled. Default `false`.
- `description` - (Optional) Free text description.
- `rule_tag` - (Optional) This can be used to specify user-controlled rule tag. If not specified, it will report rule ID after creation. Must be between 65537-131072.
- `original_address` - (Required) IP address, range or subnet. This address must be the public IP address of the edge gateway for which you are configuring the DNAT rule. In the packet being inspected, this IP address or range would be those that appear as the destination IP address of the packet. These packet destination addresses are the ones translated by this DNAT rule.
- `original_port` - (Optional) Select the port or port range that the incoming traffic uses on the edge gateway to connect to the internal network on which the virtual machines are connected. This selection is not available when the Protocol is set to `icmp` or `any`. Default `any`.
- `translated_address` - (Required) IP address, range or subnet. IP addresses to which destination addresses on inbound packets will be translated. These addresses are the IP addresses of the one or more virtual machines for which you are configuring DNAT so that they can receive traffic from the external network.

- `translated_port` - (Optional) Select the port or port range that inbound traffic is connecting to on the virtual machines on the internal network. These ports are the ones into which the DNAT rule is translating for the packets inbound to the virtual machines.
- `protocol` - (Optional) Select the protocol to which the rule applies. One of `tcp`, `udp`, `icmp`, `any`. Default `any` protocols, select Any.
- `icmp_type` - (Optional) Only when `protocol` is set to `icmp`. One of `any`, `address-mask-request`, `address-mask-reply`, `destination-unreachable`, `echo-request`, `echo-reply`, `parameter-problem`, `redirect`, `router-advertisement`, `router-solicitation`, `source-quench`, `time-exceeded`, `timestamp-request`, `timestamp-reply`. Default `any`

» Attribute Reference

The following additional attributes are exported:

- `rule_type` - Possible values - `user`, `internal_high`.

» Importing

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing dnat rule can be imported into this resource via supplying the full dot separated path for DNAT rule. An example is below:

```
terraform import vcd_nsxv_dnat.imported my-org.my-org-vdc.my-edge-gw.my-dnat-rule-id
```

NOTE: the default separator (.) can be changed using Provider.import_separator or variable VCD_IMPORT_SEPARATOR

The above would import the application rule named `my-dnat-rule-id` that is defined on edge gateway `my-edge-gw` which is configured in organization named `my-org` and vDC named `my-org-vdc`.

» vcd_nsxv_snat

Provides a vCloud Director SNAT resource for advanced edge gateways (NSX-V). This can be used to create, modify, and delete source NATs to allow vApps to send external traffic. Replaces `vcd_snat` resource.

Note: This resource requires advanced edge gateway. For non-advanced edge gateways please use the `vcd_snat` resource.

Warning: Do not use older `vcd_snat` resource with this one because it will change IDs and this resource will not be able to lookup rules.

» Example Usage

```
resource "vcd_nsxv_snat" "web" {
  org = "my-org" # Optional
  vdc = "my-vdc" # Optional

  edge_gateway = "Edge Gateway Name"
  network_type = "org"
  network_name = "my-org-network"

  original_address   = "10.10.10.15/24"
  translated_address = "78.101.10.20"
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations.
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level.
- `edge_gateway` - (Required) The name of the edge gateway on which to apply the SNAT rule.
- `network_type` - (Required) Type of the network on which to apply the DNAT rule. Possible values `org` or `ext`.
- `network_name` - (Required) The name of the network on which to apply the SNAT rule.
- `enabled` - (Optional) Defines if the rule is enabaled. Default `true`.
- `logging_enabled` - (Optional) Defines if the logging for this rule is enabaled. Default `false`.
- `description` - (Optional) Free text description.
- `rule_tag` - (Optional) This can be used to specify user-controlled rule tag. If not specified, it will report rule ID after creation. Must be between 65537-131072.
- `original_address` - (Required) IP address, range or subnet. These addresses are the IP addresses of one or more virtual machines for which you are configuring the SNAT rule so that they can send traffic to the external network.
- `translated_address` - (Required) IP address, range or subnet. This address is always the public IP address of the gateway for which you are configuring the SNAT rule. Specifies the IP address to which source addresses (the virtual machines) on outbound packets are translated to when they send traffic to the external network.

» Attribute Reference

The following additional attributes are exported:

- `rule_type` - Possible values - `user`, `internal_high`.

» Importing

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing dnat rule can be imported into this resource via supplying the full dot separated path for SNAT rule. An example is below:

```
terraform import vcd_nsxv_dnat.imported my-org.my-org-vdc.my-edge-gw.my-snat-rule-id
```

NOTE: the default separator (.) can be changed using Provider.import_separator or variable VCD_IMPORT_SEPARATOR

The above would import the application rule named `my-snat-rule-id` that is defined on edge gateway `my-edge-gw` which is configured in organization named `my-org` and vDC named `my-org-vdc`.

» vcd_nsxv_firewall_rule

Provides a vCloud Director firewall rule resource for advanced edge gateways (NSX-V). This can be used to create, modify, and delete firewall rules. Replaces `vcd_firewall_rules` resource.

Note: This resource requires advanced edge gateway (NSX-V). For non-advanced edge gateways please use the `vcd_firewall_rules` resource.

» Example Usage 1 (Minimal input with dynamic edge gateway IP)

```
data "vcd_edgegateway" "mygw" {
  org          = "my-org"
  vdc          = "my-vdc"
  name         = "my-edge-gateway-name"
}

resource "vcd_nsxv_firewall_rule" "my-rule-1" {
  org          = "my-org"
  vdc          = "my-vdc"
  edge_gateway = "my-edge-gateway"
  source {
    ip_sets = [vcd_ipset.test-ipset2.name]
  }

  destination {
    ip_addresses = ["${data.vcd_edgegateway.mygw.default_external_network_ip}"]
  }

  service {
    protocol = "any"
  }
}
```

» Example Usage 2 (Multiple services)

```
resource "vcd_nsxv_firewall_rule" "my-rule-1" {
  org          = "my-org"
  vdc          = "my-vdc"
  edge_gateway = "my-edge-gateway"

  source {
    ip_addresses       = ["any"]
    gateway_interfaces = ["internal"]
  }

  destination {
    ip_addresses = ["192.168.1.110"]
  }

  service {
    protocol = "icmp"
  }

  service {
    protocol = "tcp"
    port     = "443"
  }
}
```

» Example Usage 3 (Use exclusion in source)

```
resource "vcd_nsxv_firewall_rule" "my-rule-1" {
  org          = "my-org"
  vdc          = "my-vdc"
  edge_gateway = "my-edge-gateway"

  source {
    exclude            = true
    gateway_interfaces = ["internal"]
  }

  destination {
    ip_addresses = ["any"]
  }

  service {
    protocol = "icmp"
  }
}
```

» Example Usage 4 (Deny rule using exclusion and priority set using above_rule_id)

```
resource "vcd_nsxv_firewall_rule" "my-rule-1" {
  org          = "my-org"
  vdc          = "my-vdc"
  edge_gateway = "my-edge-gateway"

  logging_enabled = "true"
  action          = "deny"

  source {
    ip_addresses = ["30.10.10.0/24", "31.10.10.0/24"]
    org_networks = ["org-net-1", "org-net-2"]
  }

  destination {
    ip_addresses = ["any"]
  }

  service {
    protocol = "icmp"
  }
}

resource "vcd_nsxv_firewall_rule" "my-rule-2" {
  org          = "my-org"
  vdc          = "my-vdc"
  edge_gateway = "my-edge-gateway"

  # This attribute allows to ensure rule is inserted above the referred one
  # in rule processing engine
  above_rule_id = "${vcd_nsxv_firewall_rule.my-rule-1.id}"
  name          = "my-friendly-name"

  source {
    ip_addresses = ["30.10.10.0/24", "31.10.10.0/24"]
    org_networks = ["org-net-1", "org-net-2"]
  }

  destination {
    ip_addresses = ["any"]
  }

  service {
    protocol = "icmp"
  }
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations.
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level.
- `edge_gateway` - (Required) The name of the edge gateway on which to apply the firewall rule.
- `action` - (Optional) Defines if the rule is set to `accept` or `deny` traffic. Default `accept`
- `enabled` - (Optional) Defines if the rule is enabaled. Default `true`.
- `logging_enabled` - (Optional) Defines if the logging for this rule is enabaled. Default `false`.
- `name` - (Optional) Free text name. Can be duplicate.
- `rule_tag` - (Optional) This can be used to specify user-controlled rule tag. If not specified, it will report rule ID after creation. Must be between 65537-131072.
- `above_rule_id` - (Optional) This can be used to alter default rule placement order. By default every rule is appended to the end of firewall rule list. When a value of another rule is set - this rule will be placed above the specified rule.

- `source` - (Required) Exactly one block to define source criteria for firewall. See Endpoint and example for usage details.
- `destination` - (Required) Exactly one block to define source criteria for firewall. See Endpoint and example for usage details.
- `service` - (Required) One or more blocks to define protocol and port details. Use multiple blocks if you want to define multiple port/protocol combinations for the same rule. See Service and example for usage details.

» Endpoint (source or destination)

- `exclude` - (Optional) When the toggle exclusion is selected, the rule is applied to traffic on all sources except for the locations you excluded. When the toggle exclusion is not selected, the rule applies to traffic you specified. Default `false`. This example uses it.
- `ip_addresses` - (Optional) A set of IP addresses, CIDRs or ranges. A keyword `any` is also accepted as a parameter.
- `gateway_interfaces` - (Optional) A set of with either three keywords `vse` (UI names it as `any`), `internal`, `external` or an org network name. It automatically looks up vNic in the backend.
- `virtual_machine_ids` - (Optional) A set of `.id` fields of `vcd_vapp_vm` resources.
- `org_networks` - (Optional) A set of org network names.
- `ip_sets` - (Optional) A set of existing IP set names (either created manually or configured using `vcd_nsxv_ip_set` resource)

» Service

- `protocol` - (Required) One of `any`, `tcp`, `udp`, `icmp` to apply.
- `port` - (Optional) Port number or range separated by `-` for port number. Default 'any'.
- `source_port` - (Optional) Port number or range separated by `-` for port number. Default 'any'.

» Attribute Reference

The following additional attributes are exported:

- `rule_type` - Possible values - `user`, `internal_high`.

» Importing

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing firewall rule can be imported into this resource via supplying the full dot separated path for firewall rule. There are a few ways as per examples below.

NOTE: The default separator (.) can be changed using Provider.import_separator or variable VCD_IMPORT_SEPARATOR

Warning: The UI shows only firewall rule order numbers (not their real IDs). Real firewall rules have IDs with larger integer numbers like `132730`. See below for possible options to use import.

» Import by real firewall rule ID

```
terraform import vcd_nsxv_firewall_rule.imported my-org-name.my-org-vdc-name.my-edge-gw-name
```

The above would import the application rule named `my-firewall-rule-id` that is defined on edge gateway `my-edge-gw-name` which is configured in organization named `my-org-name` and vDC named `my-org-vdc-name`.

» Import by firewall rule number as shown in the UI (”No.” field)

```
terraform import vcd_nsxv_firewall_rule.imported my-org-name.my-org-vdc-name.my-edge-gw-name
```

Pay attention to the specific format of firewall rule number `ui-no.3`. The `ui-no.` flags that import must be performed by UI number of firewall rule rather than real ID.

» Listing real firewall rule IDs and their numbers

If you want to list the real IDs and firewall rule numbers there is a special command **`terraform import vcd_nsxv_firewall_rule.imported list@my-org-name.my-org-vdc-name.my-edge-gw-name`** where `my-org-name` is the organization used, `my-org-vdc-name` is vDC name and `my-edge-gw-name` is edge gateway name. The output for this command should look similar to below one:

```
$ terraform import vcd_nsxv_firewall_rule.import list@my-org-name.my-org-vdc-name.my-edge-gw
vcd_nsxv_firewall_rule.import: Importing from ID "list@my-org-name.my-org-vdc-name.my-edge-g
Retrieving all firewall rules
UI No   ID      Name                                    Action  Type
-----   --      ----                                    ------  ----
1       132589  firewall                                accept  internal_high
2       132730  My deny rule                            deny    user
3       132729  My accept rule                          accept  user
4       132588  default rule for ingress traffic        deny    default_policy
```

```
Error: Resource was not imported! Please use the above ID to format the command as:
terraform import vcd_nsxv_firewall_rule.resource-name org-name.vdc-name.edge-gw-name.firewal
```

Now to import rule with UI ID 2 (real ID 132730) one could supply this command:

```
$ terraform import vcd_nsxv_firewall_rule.import my-org-name.my-org-vdc-name.my-edge-gw-name
vcd_nsxv_firewall_rule.import: Importing from ID "my-org-name.my-org-vdc-name.my-edge-gw-nam
vcd_nsxv_firewall_rule.import: Import prepared!
  Prepared vcd_nsxv_firewall_rule for import
vcd_nsxv_firewall_rule.import: Refreshing state... [id=132730]

Import successful!

The resources that were imported are shown above. These resources are now in
your Terraform state and will henceforth be managed by Terraform.
```

» vcd_nsxv_ip_set

Provides a vCloud Director IP set resource. An IP set is a group of IP addresses that you can add as the source or destination in a firewall rule or in DHCP relay configuration.

Supported in provider *v2.6+*

» Example Usage 1

```
resource "vcd_nsxv_ip_set" "test-ipset" {
  org          = "my-org"
  vdc          = "my-org-vdc"

  name                   = "ipset-one"
  is_inheritance_allowed = false
  description            = "test-ip-set-changed-description"
  ip_addresses           = ["1.1.1.1/24","10.10.10.100-10.10.10.110"]
}
```

» Example Usage 2 (minimal example)

```
resource "vcd_nsxv_ip_set" "test-ipset" {
  name                   = "ipset-two"
  ip_addresses           = ["192.168.1.1"]
}
```

» Example Usage 3 (use IP set in firewall rules)

```
resource "vcd_nsxv_ip_set" "test-ipset" {
  org          = "my-org"
  vdc          = "my-org-vdc"

  name                   = "ipset-one"
  is_inheritance_allowed = true
  description            = "test-ip-set-changed-description"
  ip_addresses           = ["1.1.1.1/24","10.10.10.100-10.10.10.110"]
}

resource "vcd_nsxv_ip_set" "test-ipset2" {
  name                   = "ipset-two"
  ip_addresses           = ["192.168.1.1"]
}

resource "vcd_nsxv_firewall_rule" "ipsets" {
    org          = "my-org"
    vdc          = "my-org-vdc"
    edge_gateway = "my-edge-gw"

  name = "rule-with-ipsets"
    action = "accept"

    source {
        ip_sets = [vcd_nsxv_ip_set.test-ipset.name]
    }

    destination {
        ip_sets = [vcd_nsxv_ip_set.test-ipset2.name]
    }

    service {
        protocol = "any"
    }
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations

- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level
- `name` - (Required) Unique IP set name.
- `description` - (Optional) An optional description for IP set.
- `ip_addresses` - (Required) A set of IP addresses, CIDRs and ranges as strings.
- `is_inheritance_allowed` (Optional) Toggle to enable inheritance to allow visibility at underlying scopes. Default `true`

» Attribute Reference

The following attributes are exported on this resource:

- `id` - ID of IP set

» Importing

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing IP set can be imported into this resource via supplying the full dot separated path IP set. An example is below:

```
terraform import vcd_nsxv_ip_set.imported org-name.vdc-name.ipset-name
```

The above would import the IP set named `ipset-name` that is defined in org named `org-name` and vDC named `vdc-name`.

» vcd_nsxv_dhcp_relay

Provides a vCloud Director Edge Gateway DHCP relay configuration resource. The DHCP relay capability provided by NSX in vCloud Director environment allows to leverage existing DHCP infrastructure from within vCloud Director environment without any interruption to the IP address management in existing DHCP infrastructure. DHCP messages are relayed from virtual machines to the designated DHCP servers in your physical DHCP infrastructure, which allows IP addresses controlled by the NSX software to continue to be in sync with IP addresses in the rest of your DHCP-controlled environments.

Note: This resource is a ”singleton”. Because DHCP relay settings are just edge gateway properties - only one resource per Edge Gateway is useful.

Supported in provider *v2.6+*

» Example Usage 1 (Minimal configuration)

```
resource "vcd_nsxv_dhcp_relay" "relay_config" {
  org          = "my-org"
  vdc          = "my-org-vdc"
  edge_gateway = "my-edge-gw"

  ip_addresses = ["1.1.1.1"]

  relay_agent {
    network_name = vcd_network_routed.test-routed[0].name
  }
}
```

» Example Usage 2 (Example of configuration with multiple relay agents)

```
resource "vcd_nsxv_dhcp_relay" "relay_config" {
  org          = "my-org"
  vdc          = "my-org-vdc"
  edge_gateway = "my-edge-gw"

  ip_addresses = ["1.1.1.1", "2.2.2.2"]
  domain_names = ["servergroups.domainname.com", "other.domain.com"]
  ip_sets      = [vcd_nsxv_ip_set.myset1.name, vcd_nsxv_ip_set.myset2.name]

  relay_agent {
    network_name = "my-routed-network-1"
  }

  relay_agent {
    network_name        = vcd_network_routed.db-network.name
    gateway_ip_address = "10.201.1.1"
  }
}

resource "vcd_nsxv_ip_set" "myset1" {
  org          = "my-org"
  vdc          = "my-org-vdc"

  name                   = "ipset-one"
  ip_addresses           = ["10.10.10.1/24"]
}
```

```
resource "vcd_nsxv_ip_set" "myset2" {
  org          = "my-org"
  vdc          = "my-org-vdc"

  name                    = "ipset-two"
  ip_addresses            = ["20.20.20.1/24"]
}
```

» Argument Reference

The following arguments are supported:

- `org` - (Optional) The name of organization to use, optional if defined at provider level. Useful when connected as sysadmin working across different organisations.
- `vdc` - (Optional) The name of VDC to use, optional if defined at provider level.
- `edge_gateway` - (Required) The name of the edge gateway on which DHCP relay is to be configured.
- `ip_addresses` - (Optional) A set of IP addresses.
- `domain_names` - (Optional) A set of domain names.
- `ip_sets` - (Optional) A set of IP set names.
- `relay_agent` - (Required) One or more blocks to define Org network and optional IP address of edge gateway interfaces from which DHCP messages are to be relayed to the external DHCP relay server(s). See Relay Agent and example for usage details.

» Relay Agent

- `network_name` - (Required) An existing Org network name from which DHCP messages are to be relayed.
- `gateway_ip_address` - (Optional) IP address on edge gateway to be used for relaying messages. Primary address of edge gateway interface will be picked if not specified.

» Importing

Note: The current implementation of Terraform import can only import resources into the state. It does not generate configuration. More information.

An existing DHCP relay configuration can be imported into this resource via supplying the full dot separated path for your edge gateway. An example is below:

```
terraform import vcd_nsxv_dhcp_relay.imported my-org.my-org-vdc.my-edge-gw
```

The above would import the DHCP relay settings that are defined on edge gateway `my-edge-gw` which is configured in organization named `my-org` and vDC named `my-org-vdc`.