Nutanix Provider

The provider is used to interact with the many resources supported by Nutanix. The provider needs to be configured with the proper credentials before it can be used.

Use the navigation to the left to read about the available resources.

Example Usage

```
provider "nutanix" {
  username     = "xxxx"
  password     = "xxxx"
  endpoint     = "xxxx"
  insecure     = true
  port         = 9440
  wait_timeout = 10
}
```

Authentication

The Nutanix provider offers a flexible means of providing credentials for authentication. The following methods are supported, in this order, and explained below:

Static credentials

Static credentials can be provided by adding the fowlloing attributes in-line in the Nutanix provider block:

Usage:

```
provider "nutanix" {
  username     = "xxxx"
  password     = "xxxx"
  endpoint     = "xxxx"
  insecure     = true
  port         = 9440
  wait_timeout = 10 //Optional
  proxy_url    =  "xxxx" //Optional
}
```

Environment variables

You can provide your credentials via environment variables, representing your Nutanix authentication.

```
provider "nutanix" {}
```

Usage:

```
$ export NUTANIX_USERNAME="xxxx"
$ export NUTANIX_PASSWORD="xxxx"
$ export NUTANIX_INSECURE="xxxx"
$ export NUTANIX_PORT="xxxx"
$ export NUTANIX_ENDPOINT="xxxx"
$ export NUTANIX_WAIT_TIMEOUT = "xxx"
$ export NUTANIX_PROXY_URL = "xxx"

$ terraform plan
```

nutanix_category_key

Describe a Nutanix Category Key and its values (if it has them).

Example Usage

```
resource "nutanix_category_key" "test_key_value"{
    name = "data_source_category_key_test_values"
    description = "Data Source CategoryKey Test with Values"
}

resource "nutanix_category_value" "test_value"{
    name = nutanix_category_key.test_key_value.name
    value = "test_category_value_data_source"
    description = "Data Source CategoryValue Test with Values"
}


data "nutanix_category_key" "test_key_value" {
    name = nutanix_category_key.test_key_value.name
}
```

Argument Reference

The following arguments are supported:

- `name` : - (Required) The name for the category key.

Attributes Reference

The following attributes are exported:

- `system_defined` : - Specifying whether its a system defined category.
- `description` : - A description for category key.
- `api_version` - The version of the API.
- `values` : - A list of the values from this category key (if it has them).

See detailed information in Nutanix Image (https://nutanix.github.io/Automation/experimental/swagger-redoc-sandbox/#tag/category/paths/~1categories~1{name}/get).

nutanix_cluster

Describes Clusters

Example Usage

```
data "nutanix_cluster" "cluster" {
   cluster_id = "<YOUR-CLUSTER-ID>"
}`
```

Argument Reference

The following arguments are supported:

- `cluster_id` : Represents clusters uuid

Metadata

The metadata attribute exports the following:

- `last_update_time` : - UTC date and time in RFC-3339 format when image was last updated.
- `uuid` : - image uuid.
- `creation_time` : - UTC date and time in RFC-3339 format when image was created.
- `spec_version` : - Version number of the latest spec.
- `spec_hash` : - Hash of the spec. This will be returned from server.
- `name` : - image name.

Categories

The categories attribute supports the following:

- `name` : - the key name.
- `value` : - value of the key.

Attribute Reference

The following attributes are exported:

- `name` : - The name for the image.

- `categories` : - Categories for the image.
- `project_reference` : - The reference to a project.
- `owner_reference` : - The reference to a user.
- `availability_zone_reference` : - The reference to a availability_zone.
- `api_version` - The API version.
- `description` : - A description for image.
- `metadata` : - The image kind metadata.
- `state` : - The state of the cluster entity.
- `gpu_driver_version` : - GPU driver version.
- `client_auth` : - Client authentication config.
- `authorized_piblic_key_list` : - List of valid ssh keys for the cluster.
- `software_map_ncc` : - Map of software on the cluster with software type as the key.
- `software_map_nos` : - Map of software on the cluster with software type as the key.
- `encryption_status` : - Cluster encryption status.
- `ssl_key_type` : - SSL key type. Key types with RSA_2048, ECDSA_256 and ECDSA_384 are supported for key generation and importing.
- `ssl_key_signing_info` : - Customer information used in Certificate Signing Request for creating digital certificates.
- `ssl_key_expire_datetime` : - UTC date and time in RFC-3339 format when the key expires
- `service_list` : - Array of enabled cluster services. For example, a cluster can function as both AOS and cloud data gateway. - 'AOS': Regular Prism Element - 'PRISM_CENTRAL': Prism Central - 'CLOUD_DATA_GATEWAY': Cloud backup and DR gateway - 'AFS': Cluster for file server - 'WITNESS' : Witness cluster - 'XI_PORTAL': Xi cluster.
- `supported_information_verbosity` : - Verbosity level settings for populating support information. - 'Nothing': Send nothing - 'Basic': Send basic information - skip core dump and hypervisor stats information - 'BasicPlusCoreDump': Send basic and core dump information - 'All': Send all information (Default value: BASIC_PLUS_CORE_DUMP)
- `certification_signing_info` : - Customer information used in Certificate Signing Request for creating digital certificates.
- `operation_mode` : - Cluster operation mode. - 'NORMAL': Cluster is operating normally. - 'READ_ONLY': Cluster is operating in read only mode. - 'STAND_ALONE': Only one node is operational in the cluster. This is valid only for single node or two node clusters. - 'SWITCH_TO_TWO_NODE': Cluster is moving from single node to two node cluster. - 'OVERRIDE': Valid only for single node cluster. If the user wants to run vms on a single node cluster in read only mode, he can set the cluster peration mode to override. Writes will be allowed in override mode.
- `ca_certificate_list` : - Zone name used in value of TZ environment variable.
- `enabled_feature_list` : - Array of enabled features.
- `is_available` : - Indicates if cluster is available to contact. (Readonly)
- `build` : - Cluster build details.

- `timezone` : - Zone name used in value of TZ environment variable.
- `cluster_arch` : - Cluster architecture. (Readonly, Options: Options : X86_64 , PPC64LE)
- `management_server_list` : - List of cluster management servers. (Readonly)
- `masquerading_port` : - Port used together with masquerading_ip to connect to the cluster.
- `masquerading_ip` : - The cluster NAT'd or proxy IP which maps to the cluster local IP.
- `external_ip` : - The local IP of cluster visible externally.
- `http_proxy_list` : - List of proxies to connect to the service centers.
- `smtp_server_type` : - SMTP Server type.
- `smtp_server_email_address` : - SMTP Server Email Address.
- `smtp_server_credentials` : - SMTP Server Credentials.
- `smtp_server_proxy_type_list` : - SMTP Server Proxy Type List
- `smtp_server_address` : - SMTP Server Address.
- `ntp_server_ip_list` : - The list of IP addresses or FQDNs of the NTP servers.
- `external_subnet` : - External subnet for cross server communication. The format is IP/netmask. (default 172.16.0.0/255.240.0.0)
- `external_data_services_ip` : - The cluster IP address that provides external entities access to various cluster data services.
- `internal_subnet` : - The internal subnet is local to every server - its not visible outside.iSCSI requests generated internally within the appliance (by user VMs or VMFS) are sent to the internal subnet. The format is IP/netmask.
- `domain_server_nameserver` : - The IP of the nameserver that can resolve the domain name. Must set when joining the domain.
- `domain_server_name` : - Joined domain name. In 'put' request, empty name will unjoin the cluster from current domain.
- `domain_server_credentials` : - Cluster domain credentials.
- `nfs_subnet_whitelist` : - Comma separated list of subnets (of the form 'a.b.c.d/l.m.n.o') that are allowed to send NFS requests to this container. If not specified, the global NFS whitelist will be looked up for access permission. The internal subnet is always automatically considered part of the whitelist, even if the field below does not explicitly specify it. Similarly, all the hypervisor IPs are considered part of the whitelist. Finally, to permit debugging, all of the SVMs local IPs are considered to be implicitly part of the whitelist.
- `name_server_ip_list` : - The list of IP addresses of the name servers.
- `http_proxy_whitelist` : - HTTP proxy whitelist.
- `analysis_vm_efficiency_map` : - Map of cluster efficiency which includes numbers of inefficient vms. The value is populated by analytics on PC. (Readonly)

Reference

The `project_reference` , `owner_reference` , `availability_zone_reference` , attributes supports the following:

- `kind` : - The kind name (Default value: project).
- `name` : - the name.
- `uuid` : - the uuid.

Version

The version attribute supports the following:

- `product_name` : - Name of the producer/distribution of the image. For example windows or red hat.
- `product_version` : - Version string for the disk image.

See detailed information in Nutanix Image (https://nutanix.github.io/Automation/experimental/swagger-redoc-sandbox/#tag/clusters/paths/~1clusters~1multicluster_config/post).

nutanix_clusters

Describes Clusters

Example Usage

```
data "nutanix_clusters" "clusters" {}
```

Argument Reference

No arguments are supported:

Metadata

The metadata attribute exports the following:

- `last_update_time` : - UTC date and time in RFC-3339 format when image was last updated.
- `uuid` : - image UUID.
- `creation_time` : - UTC date and time in RFC-3339 format when image was created.
- `spec_version` : - Version number of the latest spec.
- `spec_hash` : - Hash of the spec. This will be returned from server.
- `name` : - image name.

Categories

The categories attribute supports the following:

- `name` : - the key name.
- `value` : - value of the key.

Attribute Reference

The following attributes are exported:

- `entities` : List of Clusters

Entities

The entities attribute element contains the followings attributes:

- `name` : - The name for the image.
- `categories` : - Categories for the image.
- `project_reference` : - The reference to a project.
- `owner_reference` : - The reference to a user.
- `availability_zone_reference` : - The reference to a availability_zone.
- `api_version` - The API version.
- `description` : - A description for image.
- `metadata` : - The image kind metadata.
- `state` : - The state of the cluster entity.
- `gpu_driver_version` : - GPU driver version.
- `client_auth` : - Client authentication config.
- `authorized_piblic_key_list` : - List of valid ssh keys for the cluster.
- `software_map_ncc` : - Map of software on the cluster with software type as the key.
- `software_map_nos` : - Map of software on the cluster with software type as the key.
- `encryption_status` : - Cluster encryption status.
- `ssl_key_type` : - SSL key type. Key types with RSA_2048, ECDSA_256 and ECDSA_384 are supported for key generation and importing.
- `ssl_key_signing_info` : - Customer information used in Certificate Signing Request for creating digital certificates.
- `ssl_key_expire_datetime` : - UTC date and time in RFC-3339 format when the key expires
- `service_list` : - Array of enabled cluster services. For example, a cluster can function as both AOS and cloud data gateway. - 'AOS': Regular Prism Element - 'PRISM_CENTRAL': Prism Central - 'CLOUD_DATA_GATEWAY': Cloud backup and DR gateway - 'AFS': Cluster for file server - 'WITNESS' : Witness cluster - 'XI_PORTAL': Xi cluster.
- `supported_information_verbosity` : - Verbosity level settings for populating support information. - 'Nothing': Send nothing - 'Basic': Send basic information - skip core dump and hypervisor stats information - 'BasicPlusCoreDump': Send basic and core dump information - 'All': Send all information (Default value: BASIC_PLUS_CORE_DUMP)
- `certification_signing_info` : - Customer information used in Certificate Signing Request for creating digital certificates.
- `operation_mode` : - Cluster operation mode. - 'NORMAL': Cluster is operating normally. - 'READ_ONLY': Cluster is operating in read only mode. - 'STAND_ALONE': Only one node is operational in the cluster. This is valid only for single node or two node clusters. - 'SWITCH_TO_TWO_NODE': Cluster is moving from single node to two node cluster. - 'OVERRIDE': Valid only for single node cluster. If the user wants to run vms on a single node cluster in read only mode, he can set the cluster peration mode to override. Writes will be allowed in override mode.
- `ca_certificate_list` : - Zone name used in value of TZ environment variable.
- `enabled_feature_list` : - Array of enabled features.
- `is_available` : - Indicates if cluster is available to contact. (Readonly)

- `build` : - Cluster build details.
- `timezone` : - Zone name used in value of TZ environment variable.
- `cluster_arch` : - Cluster architecture. (Readonly, Options: Options : X86_64 , PPC64LE)
- `management_server_list` : - List of cluster management servers. (Readonly)
- `masquerading_port` : - Port used together with masquerading_ip to connect to the cluster.
- `masquerading_ip` : - The cluster NAT'd or proxy IP which maps to the cluster local IP.
- `external_ip` : - The local IP of cluster visible externally.
- `http_proxy_list` : - List of proxies to connect to the service centers.
- `smtp_server_type` : - SMTP Server type.
- `smtp_server_email_address` : - SMTP Server Email Address.
- `smtp_server_credentials` : - SMTP Server Credentials.
- `smtp_server_proxy_type_list` : - SMTP Server Proxy Type List
- `smtp_server_address` : - SMTP Server Address.
- `ntp_server_ip_list` : - The list of IP addresses or FQDNs of the NTP servers.
- `external_subnet` : - External subnet for cross server communication. The format is IP/netmask. (default 172.16.0.0/255.240.0.0)
- `external_data_services_ip` : - The cluster IP address that provides external entities access to various cluster data services.
- `internal_subnet` : - The internal subnet is local to every server - its not visible outside.iSCSI requests generated internally within the appliance (by user VMs or VMFS) are sent to the internal subnet. The format is IP/netmask.
- `domain_server_nameserver` : - The IP of the nameserver that can resolve the domain name. Must set when joining the domain.
- `domain_server_name` : - Joined domain name. In 'put' request, empty name will unjoin the cluster from current domain.
- `domain_server_credentials` : - Cluster domain credentials.
- `nfs_subnet_whitelist` : - Comma separated list of subnets (of the form 'a.b.c.d/l.m.n.o') that are allowed to send NFS requests to this container. If not specified, the global NFS whitelist will be looked up for access permission. The internal subnet is always automatically considered part of the whitelist, even if the field below does not explicitly specify it. Similarly, all the hypervisor IPs are considered part of the whitelist. Finally, to permit debugging, all of the SVMs local IPs are considered to be implicitly part of the whitelist.
- `name_server_ip_list` : - The list of IP addresses of the name servers.
- `http_proxy_whitelist` : - HTTP proxy whitelist.
- `analysis_vm_efficiency_map` : - Map of cluster efficiency which includes numbers of inefficient vms. The value is populated by analytics on PC. (Readonly)

Reference

The `project_reference`, `owner_reference`, `availability_zone_reference`, `cluster_reference`, attributes supports the following:

- `kind` : - The kind name (Default value: project).
- `name` : - the name.
- `UUID` : - the UUID.

Version

The version attribute supports the following:

- `product_name` : - Name of the producer/distribution of the image. For example windows or red hat.
- `product_version` : - Version string for the disk image.

See detailed information in Nutanix Image (https://nutanix.github.io/Automation/experimental/swagger-redoc-sandbox/#tag/clusters/paths/~1clusters~1multicluster_config/post).

nutanix_image

Describes a Image

Example Usage

```
resource "nutanix_image" "test" {
  name        = "Ubuntu"
  description = "Ubuntu"
  source_uri  = "http://archive.ubuntu.com/ubuntu/dists/bionic/main/installer-amd64/current/images/netboo
t/mini.iso"
}


data "nutanix_image" "test" {
    image_id = nutanix_image.test.id
}

data "nutanix_image" "testname" {
    image_name = nutanix_image.test.name
}
```

Argument Reference

The following arguments are supported:

- `image_id` : Represents image UUID
- `image_name` : Represents image name

Attribute Reference

The following attributes are exported:

- `name` : - The name for the image.
- `categories` : - Categories for the image.
- `project_reference` : - The reference to a project.
- `owner_reference` : - The reference to a user.
- `availability_zone_reference` : - The reference to a availability_zone.
- `api_version` -
- `description` : - A description for image.
- `architecture` : - The supported CPU architecture for a disk image.

- `checksum` : - Checksum of the image. The checksum is used for image validation if the image has a source specified. For images that do not have their source specified the checksum is generated by the image service.
- `image_type` : - The type of image.
- `source_uri` : - The source URI points at the location of the source image which is used to create/update image.
- `version` : - The image version.
- `metadata` : - The image kind metadata.
- `retrieval_uri_list` : - List of URIs where the raw image data can be accessed.
- `size_bytes` : - The size of the image in bytes.
- `state` : - The state of the image.

Metadata

The metadata attribute exports the following:

- `last_update_time` : - UTC date and time in RFC-3339 format when image was last updated.
- `UUID` : - image UUID.
- `creation_time` : - UTC date and time in RFC-3339 format when image was created.
- `spec_version` : - Version number of the latest spec.
- `spec_hash` : - Hash of the spec. This will be returned from server.
- `name` : - image name.

Reference

The `project_reference` , `owner_reference` , `availability_zone_reference` , `cluster_reference` , attributes supports the following:

- `kind` : - The kind name (Default value: project).
- `name` : - the name.
- `uuid` : - the UUID.

Version

The version attribute supports the following:

- `product_name` : - Name of the producer/distribution of the image. For example windows or red hat.
- `product_version` : - Version string for the disk image.

See detailed information in Nutanix Image (http://developer.nutanix.com/reference/prism_central/v3/#images).

nutanix_network_security_rule

Describes a Network security rule

Example Usage (Isolate Development VMs from Production VMs and get its information)

```
resource "nutanix_network_security_rule" "isolation" {
    name        = "example-isolation-rule"
    description = "Isolation Rule Example"

    isolation_rule_action = "APPLY"

    isolation_rule_first_entity_filter_kind_list = ["vm"]
    isolation_rule_first_entity_filter_type      = "CATEGORIES_MATCH_ALL"
    isolation_rule_first_entity_filter_params {
        name   = "Environment"
        values = ["Dev"]
    }

    isolation_rule_second_entity_filter_kind_list = ["vm"]
    isolation_rule_second_entity_filter_type      = "CATEGORIES_MATCH_ALL"
    isolation_rule_second_entity_filter_params {
        name   = "Environment"
        values = ["Production"]
    }
}

data "nutanix_network_security_rule" "test" {
  network_security_rule_id = "${nutanix_network_security_rule.isolation.id}"
}
```

Argument Reference

The following arguments are supported:

- `network_security_rule_id` : Represents network security rule UUID

Attribute Reference

The following attributes are exported:

The following arguments are supported:

- `name` : - (Required) The name for the image.
- `categories` : - (Optional) Categories for the image.

- `project_reference` : - (Optional) The reference to a project.
- `owner_reference` : - (Optional) The reference to a user.
- `api_version` - (Optional)
- `description` : - (Optional) A description for image.
- `quarantine_rule_action` : - (Optional) These rules are used for quarantining suspected VMs. Target group is a required attribute. Empty inbound_allow_list will not allow anything into target group. Empty outbound_allow_list will allow everything from target group.
- `quarantine_rule_outbound_allow_list` : - (Optional)
- `quarantine_rule_target_group_default_internal_policy` : - (Optional) - Default policy for communication within target group.
- `quarantine_rule_target_group_peer_specification_type` : - (Optional) - Way to identify the object for which rule is applied.
- `quarantine_rule_target_group_filter_kind_list` : - (Optional) - List of kinds associated with this filter.
- `quarantine_rule_target_group_filter_type` : - (Optional) - The type of the filter being used.
- `quarantine_rule_target_group_filter_params` : - (Optional) - A list of category key and list of values.
- `quarantine_rule_inbound_allow_list` : - (Optional)
- `app_rule_action` : - (Optional) - These rules govern what flows are allowed. Target group is a required attribute. Empty inbound_allow_list will not anything into target group. Empty outbound_allow_list will allow everything from target group.
- `app_rule_outbound_allow_list` : - (Optional)
- `app_rule_target_group_default_internal_policy` : - (Optional) - Default policy for communication within target group.
- `app_rule_target_group_peer_specification_type` : - (Optional) - Way to identify the object for which rule is applied.
- `app_rule_target_group_filter_kind_list` : - (Optional) - List of kinds associated with this filter.
- `app_rule_target_group_filter_type` : - (Optional) - The type of the filter being used.
- `app_rule_target_group_filter_params` : - (Optional) - A list of category key and list of values.
- `app_rule_inbound_allow_list` : - (Optional)
- `isolation_rule_action` : - (Optional) - These rules are used for environmental isolation.
- `app_rule_inbound_allow_list` : - (Optional)
- `isolation_rule_first_entity_filter_kind_list` : - (Optional) - List of kinds associated with this filter.
- `isolation_rule_first_entity_filter_type` : - (Optional) - The type of the filter being used.
- `isolation_rule_first_entity_filter_params` : - (Optional) - A list of category key and list of values.
- `isolation_rule_second_entity_filter_kind_list` : - (Optional) - List of kinds associated with this filter.

- `isolation_rule_second_entity_filter_type` : - (Optional) - The type of the filter being used.
- `isolation_rule_second_entity_filter_params` : - (Optional) - A list of category key and list of values.

Metadata

The metadata attribute exports the following:

- `last_update_time` : - UTC date and time in RFC-3339 format when image was last updated.
- `UUID` : - image UUID.
- `creation_time` : - UTC date and time in RFC-3339 format when image was created.
- `spec_version` : - Version number of the latest spec.
- `spec_hash` : - Hash of the spec. This will be returned from server.
- `name` : - image name.

Categories

The categories attribute supports the following:

- `name` : - the key name.
- `value` : - value of the key.

Reference

The `project_reference` , `owner_reference` , `availability_zone_reference` , `cluster_reference` , attributes supports the following:

- `kind` : - The kind name (Default value: project).
- `name` : - the name.
- `uuid` : - the UUID.

Version

The version attribute supports the following:

- `product_name` : - Name of the producer/distribution of the image. For example windows or red hat.
- `product_version` : - Version string for the disk image.

See detailed information in Nutanix Security Rules (https://www.nutanix.dev/reference/prism_central/v3/api/network-security-rules/getnetworksecurityrulesuuid).

nutanix_subnet

Provides a resource to create a subnet based on the input parameters. A subnet is a block of IP addresses.

Example Usage

```
data "nutanix_clusters" "clusters" {
  metadata = {
    length = 2
  }
}

output "cluster" {
  value = data.nutanix_clusters.clusters.entities.0.metadata.uuid
}

resource "nutanix_subnet" "test" {
  name = "example-subnet"

  cluster_reference = {
  kind = "cluster"
  UUID = data.nutanix_clusters.clusters.entities.0.metadata.uuid
}

  vlan_id = 201
  subnet_type = "VLAN"

  prefix_length = 24
  default_gateway_ip = "192.168.0.1"
  subnet_ip = "192.168.0.0"
  #ip_config_pool_list_ranges = ["192.168.0.5", "192.168.0.100"]

  dhcp_options {
    boot_file_name = "bootfile"
    tftp_server_name = "192.168.0.252"
    domain_name = "nutanix"
  }

  dhcp_domain_name_server_list = ["8.8.8.8", "4.2.2.2"]
  dhcp_domain_search_list = ["nutanix.com", "calm.io"]
}

data "nutanix_subnet" "test" {
  subnet_id = nutanix_subnet.test.id
}

data "nutanix_subnet" "test-name" {
  subnet_name = nutanix_subnet.test.name
}
```

Argument Reference

The following arguments are supported:

- `subnet_id` : - (Optional) The ID for the subnet.
- `subnet_name` : - (Optional) The name for the subnet

Attributes Reference

The following attributes are exported:

- `metadata` : - (Required) The subnet kind metadata.
- `availability_zone_reference` : - (Optional) The reference to a availability_zone.
- `cluster_reference` : - (Optional) The reference to a cluster.
- `cluster_name` : - (Optional) The name of a cluster.
- `description` : - (Optional) A description for subnet.
- `name` : - (Optional) Subnet name (Readonly).
- `categories` : - (Optional) The API Version.
- `owner_reference` : - (Optional) The reference to a user.
- `project_reference` : - (Optional) The reference to a project.
- `vswitch_name` : - (Optional).
- `subnet_type` : - (Optional).
- `default_gateway_ip` : - (Optional) Default gateway IP address.
- `prefix_length` : - (Optional). IP prefix length of the Subnet.
- `subnet_ip` : - (Optional) Subnet IP address.
- `dhcp_server_address` : - (Optional) Host address.
- `dhcp_server_address_port` : - (Optional) Port Number.
- `dhcp_options` : - (Optional) Spec for defining DHCP options.
- `dhcp_domain_search_list` : - (Optional).
- `vlan_id` : - (Optional).
- `network_function_chain_reference` : - (Optional) The reference to a network_function_chain.
- `state` : - The state of the subnet.

Metadata

The metadata attribute exports the following:

- `last_update_time` : - UTC date and time in RFC-3339 format when subnet was last updated.

- `UUID` : - subnet UUID.
- `creation_time` : - UTC date and time in RFC-3339 format when subnet was created.
- `spec_version` : - Version number of the latest spec.
- `spec_hash` : - Hash of the spec. This will be returned from server.
- `name` : - subnet name.

Categories

The categories attribute supports the following:

- `name` : - the key name.
- `value` : - value of the key.

Reference

The `project_reference`, `owner_reference`, `availability_zone_reference`, `cluster_reference`, `network_function_chain_reference`, `subnet_reference`.

attributes supports the following:

- `kind` : - The kind name (Default value: project)(Required).
- `name` : - the name(Optional).
- `uuid` : - the UUID(Required).

Note: `cluster_reference`, `subnet_reference` does not support the attribute `name`

See detailed information in Nutanix Subnet (http://developer.nutanix.com/reference/prism_central/v3/#definitions-subnet_resources).

nutanix_virtual_machine

Describes a Virtual Machine

Example Usage

```
data "nutanix_clusters" "clusters" {
 metadata = {
  length = 2
 }
}

output "cluster" {
 value = data.nutanix_clusters.clusters.entities.0.metadata.uuid
}

resource "nutanix_virtual_machine" "vm1" {
 name = "test-dou-%d"
 cluster_uuid= data.nutanix_clusters.clusters.entities.0.metadata.uuid

 num_vcpus_per_socket = 1
 num_sockets     = 1
 memory_size_mib   = 2048
 power_state     = "ON"
}

data "nutanix_virtual_machine" "nutanix_virtual_machine" {
 vm_id = nutanix_virtual_machine.vm1.id
}
```

Argument Reference

The following arguments are supported:

- `vm_id` : Represents virtual machine UUID

Attribute Reference

The following attributes are exported:

- `name` : - The name for the vm.
- `cluster_reference` : - The reference to a cluster.
- `cluster_name` : - The name of the reference to the cluster.
- `categories` : - Categories for the vm.
- `project_reference` : - The reference to a project.

- `owner_reference` : - The reference to a user.
- `availability_zone_reference` : - The reference to a availability_zone.
- `api_version` - The version of the API.
- `description` : - A description for vm.
- `num_vnuma_nodes` : - Number of vNUMA nodes. 0 means vNUMA is disabled.
- `nic_list` : - NICs attached to the VM.
- `serial_port_list` : - (Optional) Serial Ports configured on the VM.
- `guest_os_id` : - Guest OS Identifier. For ESX, refer to VMware documentation link (https://www.vmware.com/support/developer/converter-sdk/conv43_apireference/vim.vm.GuestOsDescriptor.GuestOsIdentifier.html) for the list of guest OS identifiers.
- `power_state` : - The current or desired power state of the VM. (Options : ON , OFF)
- `nutanix_guest_tools` : - Information regarding Nutanix Guest Tools.
- `ngt_credentials` : - Credentials to login server.
- `ngt_enabled_capability_list` - Application names that are enabled.
- `num_vcpus_per_socket` : - Number of vCPUs per socket.
- `num_sockets` : - Number of vCPU sockets.
- `gpu_list` : - GPUs attached to the VM.
- `parent_referece` : - Reference to an entity that the VM cloned from.
- `memory_size_mib` : - Memory size in MiB.
- `boot_device_order_list` : - Indicates the order of device types in which VM should try to boot from. If boot device order is not provided the system will decide appropriate boot device order.
- `boot_device_disk_address` : - Address of disk to boot from.
- `boot_device_mac_address` : - MAC address of nic to boot from.
- `hardware_clock_timezone` : - VM's hardware clock timezone in IANA TZDB format (America/Los_Angeles).
- `guest_customization_cloud_init_user_data` : - The contents of the user_data configuration for cloud-init. This can be formatted as YAML, JSON, or could be a shell script. The value must be base64 encoded.
- `guest_customization_cloud_init_meta_data` - The contents of the meta_data configuration for cloud-init. This can be formatted as YAML or JSON. The value must be base64 encoded.
- `guest_customization_is_overridable` : - Flag to allow override of customization by deployer.
- `guest_customization_cloud_init_custom_key_values` : - Generic key value pair used for custom attributes in cloud init.
- `guest_customization_sysprep` : - VM guests may be customized at boot time using one of several different methods. Currently, cloud-init w/ ConfigDriveV2 (for Linux VMs) and Sysprep (for Windows VMs) are supported. Only ONE OF sysprep or cloud_init should be provided. Note that guest customization can currently only be set during VM creation.

Attempting to change it after creation will result in an error. Additional properties can be specified. For example - in the context of VM template creation if \"override_script\" is set to \"True\" then the deployer can upload their own custom script.

- `guest_customization_sysrep_custom_key_values` : - Generic key value pair used for custom attributes in sysrep.
- `should_fail_on_script_failure` : - Extra configs related to power state transition. Indicates whether to abort ngt shutdown/reboot if script fails.
- `enable_script_exec` : - Extra configs related to power state transition. Indicates whether to execute set script before ngt shutdown/reboot.
- `power_state_mechanism` : - Indicates the mechanism guiding the VM power state transition. Currently used for the transition to \"OFF\" state. Power state mechanism (ACPI/GUEST/HARD).
- `vga_console_enabled` : - Indicates whether VGA console should be enabled or not.
- `disk_list` Disks attached to the VM.
- `metadata` : - The vm kind metadata.
- `state` : - The state of the vm.
- `ip_address` : - An IP address.
- `host_reference` : - Reference to a host.
- `hypervisor_type` : - The hypervisor type for the hypervisor the VM is hosted on.

Disk List

The disk_list attribute supports the following:

- `UUID` : - The device ID which is used to uniquely identify this particular disk.
- `disk_size_bytes` - Size of the disk in Bytes.
- `disk_size_mib` - Size of the disk in MiB. Must match the size specified in 'disk_size_bytes' - rounded up to the nearest MiB - when that field is present.
- `device_properties` - Properties to a device.
- `data_source_reference` - Reference to a data source.
- `volume_group_reference` - Reference to a volume group.

Device Properties

The device_properties attribute supports the following.

- `device_type` : - A Disk type (default: DISK).
- `disk_address` : - Address of disk to boot from.

Sysprep

The guest_customization_sysprep attribute supports the following:

- `install_type` : - Whether the guest will be freshly installed using this unattend configuration, or whether this unattend configuration will be applied to a pre-prepared image. Default is \"PREPARED\".
- `unattend_xml` : - Generic key value pair used for custom attributes.

Disk Address

The boot_device_disk_address attribute supports the following:

- `device_index` : - The index of the disk address.
- `adapter_type` : - The adapter type of the disk address.

GPU List

The gpu_list attribute supports the following:

- `frame_buffer_size_mib` : - GPU frame buffer size in MiB.
- `vendor` : - The vendor of the GPU.
- `UUID` : - UUID of the GPU.
- `name` : - Name of the GPU resource.
- `pci_address` - GPU {segment:bus:device:function} (sbdf) address if assigned.
- `fraction` - Fraction of the physical GPU assigned.
- `mode` : - The mode of this GPU.
- `num_virtual_display_heads` : - Number of supported virtual display heads.
- `guest_driver_version` : - Last determined guest driver version.
- `device_id` : - (Computed) The device ID of the GPU.

Nutanix Guest Tools

The nutanix_guest_tools attribute supports the following:

- `state` : - Nutanix Guest Tools is enabled or not.
- `ngt_state` : - Nutanix Guest Tools is enabled or not.
- `iso_mount_state` : - Desired mount state of Nutanix Guest Tools ISO.
- `version` : - Version of Nutanix Guest Tools installed on the VM.
- `available_version` : - Version of Nutanix Guest Tools available on the cluster.

- `guest_os_version` : - Version of the operating system on the VM.
- `vss_snapshot_capable` : - Whether the VM is configured to take VSS snapshots through NGT.
- `is_reachable` : - Communication from VM to CVM is active or not.
- `vm_mobility_drivers_installed` : - Whether VM mobility drivers are installed in the VM.

NIC List

The nic_list attribute supports the following:

- `nic_type` : - The type of this NIC. Defaults to NORMAL_NIC. (Options : NORMAL_NIC , DIRECT_NIC , NETWORK_FUNCTION_NIC).
- `uuid` : - The NIC's UUID, which is used to uniquely identify this particular NIC. This UUID may be used to refer to the NIC outside the context of the particular VM it is attached to.
- `floating_ip` : - The Floating IP associated with the vnic.
- `model` : - The model of this NIC. (Options : VIRTIO , E1000).
- `network_function_nic_type` : - The type of this Network function NIC. Defaults to INGRESS. (Options : INGRESS , EGRESS , TAP).
- `mac_address` : - The MAC address for the adapter.
- `ip_endpoint_list` : - IP endpoints for the adapter. Currently, IPv4 addresses are supported.
- `network_function_chain_reference` : - The reference to a network_function_chain.
- `subnet_uuid` : - The reference to a subnet.
- `subnet_name` : - The name of the subnet reference to.

Serial Port List

The `serial_port_list` attribute supports the following:

- `index` : - Index of the serial port (int).
- `is_connected` : - Indicates whether the serial port connection is connected or not (`true` or `false`).

ip_endpoint_list

The following attributes are exported:

- `ip` : - Address string.
- `type` : - Address type. It can only be "ASSIGNED" in the spec. If no type is specified in the spec, the default type is set to "ASSIGNED". (Options : ASSIGNED , LEARNED)

Metadata

The metadata attribute exports the following:

- `last_update_time` : - UTC date and time in RFC-3339 format when vm was last updated.
- `UUID` : - vm UUID.
- `creation_time` : - UTC date and time in RFC-3339 format when vm was created.
- `spec_version` : - Version number of the latest spec.
- `spec_hash` : - Hash of the spec. This will be returned from server.
- `name` : - vm name.

Categories

The categories attribute supports the following:

- `name` : - the key name.
- `value` : - value of the key.

Reference

The `project_reference`, `owner_reference`, `availability_zone_reference`, `network_function_chain_reference`, `data_source_reference`, `volume_group_reference` attributes supports the following:

- `kind` : - The kind name (Default value: project).
- `name` : - the name.
- `uud` : - the UUID.

See detailed information in Nutanix Virtual Machine (http://developer.nutanix.com/reference/prism_central/v3/#vms).

nutanix_volume_group

Describes a Volume Group

Example Usage

```
resource "nutanix_volume_group" "test" {
  name        = "VG Test"
  description = "VG Test Description"

}

data "nutanix_volume_group" "test" {
    volume_group_id = "${nutanix_volume_group.test.id}"
}
```

Argument Reference

The following arguments are supported:

- `volume_group_id` : Represents volume group UUID

Attribute Reference

The following attributes are exported:

- `name` : - (Required) The name for the volume_group.
- `categories` : - (Optional) Categories for the volume_group.
- `project_reference` : - (Optional) The reference to a project.
- `owner_reference` : - (Optional) The reference to a user.
- `api_version` - (Optional) Version of the API.
- `description` : - (Optional) A description for volume_group.
- `flash_mode` : - (Optional) Flash Mode, if enabled all volume disks of the VG will be pinned to SSD tier.
- `file_system_type` : - (Optional) File system to be used for volume.
- `sharing_status` : - (Optiional) Whether the volume group can be shared across multiple iSCSI initiators.
- `attachment_list` : - (Optional) VMs attached to volume group.
- `disk_list` : - (Optional) Volume group disk specification.
- `iscsi_target_prefix` : - (Optional) iSCSI target prefix-name.

- `metadata` : - The volume_group kind metadata.
- `state` : - The state of the Volume Group.

Disk List

The disk_list attribute supports the following:

- `vmdisk_UUID` : - (Optional) The UUID of this volume disk.
- `index` : - (Optional) Index of the volume disk in the group.
- `data_source_reference` : - (Optional) Reference to a kind
- `disk_size_mib` : - (Optional) Size of the disk in MiB.
- `storage_container_UUID` : - (Optional) Container UUID on which to create the disk.

Attachment List

The attachment_list attribute supports the following:

- `vm_reference` : - (Optional) Reference to a kind.
- `iscsi_initiator_name` : - (Optional) Name of the iSCSI initiator of the workload outside Nutanix cluster.

Metadata

The metadata attribute exports the following:

- `last_update_time` : - UTC date and time in RFC-3339 format when volume_group was last updated.
- `UUID` : - volume_group UUID.
- `creation_time` : - UTC date and time in RFC-3339 format when volume_group was created.
- `spec_version` : - Version number of the latest spec.
- `spec_hash` : - Hash of the spec. This will be returned from server.
- `name` : - volume_group name.

Reference

The `project_reference` , `owner_reference` , `availability_zone_reference` , `cluster_reference` , attributes supports the following:

- `kind` : - The kind name (Default value: project)(Required).
- `name` : - the name(Optional).
- `UUID` : - the UUID(Required).

Note: `vm_reference` and `data_source_reference` don't support `name` argument.

See detailed information in Nutanix Volume Group (https://nutanix.github.io/Automation/experimental/swagger-redoc-sandbox/#tag/volume_group).

nutanix_category_key

Provides a Nutanix Category key resource to Create a category key name.

Example Usage

```
resource "nutanix_category_key" "test"{
    name = "app-support-example"
    description = "App Support Category Key"
}
```

Argument Reference

The following arguments are supported:

- `name` : - (Required) The name for the category key.
- `description` : - (Optional) A description for category key.

Attributes Reference

The following attributes are exported:

- `system_defined` : - Specifying whether its a system defined category.
- `api_version` - (Optional) The version of the API.

See detailed information in Nutanix Image (https://nutanix.github.io/Automation/experimental/swagger-redoc-sandbox/#tag/category/paths/~1categories~1{name}/get).

nutanix_category_value

Provides a Nutanix Category value resource to Create a category value.

Example Usage

```
resource "nutanix_category_key" "test-category-key"{
    name = "app-support-1"
    description = "App Support Category Key"
}


resource "nutanix_category_value" "test"{
    name = nutanix_category_key.test-category-key.id
    description = "Test Category Value"
    value = "test-value"
}
```

Argument Reference

The following arguments are supported:

- `name` : - (Required) The category_key name for the category value.
- `value` - (Required) The value for the category value.
- `description` : - (Optional) A description for category value.

Attributes Reference

The following attributes are exported:

- `system_defined` : - Specifying whether its a system defined category.
- `api_version` - (Optional) The version of the API.

See detailed information in Nutanix Image (http://developer.nutanix.com/reference/prism_central/v3/#category).

nutanix_image

Provides a Nutanix Image resource to Create a image.

Example Usage

```
resource "nutanix_image" "test" {
  name        = "Ubuntu"
  description = "Ubuntu"
  source_uri  = "http://archive.ubuntu.com/ubuntu/dists/bionic/main/installer-amd64/current/images/netboo
t/mini.iso"
}
```

Argument Reference

The following arguments are supported:

- `name` : - (Required) The name for the image.
- `categories` : - (Optional) Categories for the image.
- `project_reference` : - (Optional) The reference to a project.
- `owner_reference` : - (Optional) The reference to a user.
- `availability_zone_reference` : - (Optional) The reference to a availability_zone.
- `description` : - (Optional) A description for image.
- `architecture` : - (Optional) The supported CPU architecture for a disk image.
- `checksum` : - (Optional) Checksum of the image. The checksum is used for image validation if the image has a source specified. For images that do not have their source specified the checksum is generated by the image service.
- `image_type` : - (Optional) The type of image.
- `source_uri` : - (Optional) The source URI points at the location of the source image which is used to create/update image.
- `source_path` : - (Optional) A local path to upload an image.
- `version` : - (Optional) The image version.

Version

The version attribute supports the following:

- `product_name` : - (Optional) Name of the producer/distribution of the image. For example windows or red hat.
- `product_version` : - (Optional) Version string for the disk image.

Attributes Reference

The following attributes are exported:

- `metadata` : - The image kind metadata.
- `retrieval_uri_list` : - List of URIs where the raw image data can be accessed.
- `size_bytes` : - The size of the image in bytes.
- `state` : - The state of the image.
- `api_version` - The version of the API.

Metadata

The metadata attribute exports the following:

- `last_update_time` : - UTC date and time in RFC-3339 format when image was last updated.
- `uuid` : - image UUID.
- `creation_time` : - UTC date and time in RFC-3339 format when image was created.
- `spec_version` : - Version number of the latest spec.
- `spec_hash` : - Hash of the spec. This will be returned from server.
- `name` : - image name.

Reference

The `project_reference`, `owner_reference`, `availability_zone_reference`, attributes supports the following:

- `kind` : - The kind name (Default value: project)(Required).
- `name` : - the name(Optional).
- `uuid` : - the UUID(Required).

See detailed information in Nutanix Image (http://developer.nutanix.com/reference/prism_central/v3/#images).

nutanix_network_security_rule

Provides a Nutanix network security rule resource to Create a network security rule.

Example Usage

Isolation Rule Example

```
resource "nutanix_network_security_rule" "isolation" {
    name        = "example-isolation-rule"
    description = "Isolation Rule Example"

    isolation_rule_action = "APPLY"

    isolation_rule_first_entity_filter_kind_list = ["vm"]
    isolation_rule_first_entity_filter_type      = "CATEGORIES_MATCH_ALL"
    isolation_rule_first_entity_filter_params {
        name   = "Environment"
        values = ["Dev"]
    }

    isolation_rule_second_entity_filter_kind_list = ["vm"]
    isolation_rule_second_entity_filter_type      = "CATEGORIES_MATCH_ALL"
    isolation_rule_second_entity_filter_params {
        name   = "Environment"
        values = ["Production"]
    }
}
```

App Rule Example with associated VMs.

```
data "nutanix_clusters" "clusters" {}

locals {
  cluster_uuid = [
    for cluster in data.nutanix_clusters.clusters.entities :
    cluster.metadata.uuid if cluster.service_list[0] != "PRISM_CENTRAL"
  ][0]
}

//Create categories.
resource "nutanix_category_key" "test-category-key" {
  name        = "TIER-1"
  description = "TIER Category Key"
}

resource "nutanix_category_key" "USER" {
  name        = "user"
  description = "user Category Key"
}
```

```
}


resource "nutanix_category_value" "WEB" {
  name        = "${nutanix_category_key.test-category-key.id}"
  description = "WEB Category Value"
  value       = "WEB-1"
}

resource "nutanix_category_value" "APP" {
  name        = "${nutanix_category_key.test-category-key.id}"
  description = "APP Category Value"
  value       = "APP-1"
}

resource "nutanix_category_value" "DB" {
  name        = "${nutanix_category_key.test-category-key.id}"
  description = "DB Category Value"
  value       = "DB-1"
}

resource "nutanix_category_value" "group" {
  name        = "${nutanix_category_key.USER.id}"
  description = "group Category Value"
  value       = "group-1"
}


//Create a cirros image
resource "nutanix_image" "cirros-034-disk" {
  name        = "test-image-vm-create-flow"
  source_uri  = "http://download.cirros-cloud.net/0.4.0/cirros-0.4.0-x86_64-disk.img"
  description = "heres a tiny linux image, not an iso, but a real disk!"
}

//APP-1 VM.
resource "nutanix_virtual_machine" "vm-app" {
  name                 = "test-dou-vm-flow-APP-1"
  cluster_uuid         = local.cluster_uuid
  num_vcpus_per_socket = 1
  num_sockets          = 1
  memory_size_mib      = 186

  nic_list {
    subnet_uuid = "c56b535c-8aff-4435-ae85-78e64a07f76d"
  }

  disk_list {
    data_source_reference = {
      kind = "image"
      uuid = nutanix_image.cirros-034-disk.id
    }

    device_properties {
      disk_address = {
        device_index = 0
        adapter_type = "SCSI"
      }
      device_type = "DISK"
    }
```

```
  }

  categories {
    name  = "Environment"
    value = "Staging"
  }

  categories {
    name  = "TIER-1"
    value = nutanix_category_value.APP.id
  }
}

#WEB-1 VM
resource "nutanix_virtual_machine" "vm-web" {
  name                 = "test-dou-vm-flow-WEB-1"
  cluster_uuid         = local.cluster_uuid
  num_vcpus_per_socket = 1
  num_sockets          = 1
  memory_size_mib      = 186

  nic_list {
    subnet_uuid = "c56b535c-8aff-4435-ae85-78e64a07f76d"
  }

  disk_list {
    data_source_reference = {
      kind = "image"
      uuid = nutanix_image.cirros-034-disk.id
    }

    device_properties {
      disk_address = {
        device_index = 0
        adapter_type = "SCSI"
      }
      device_type = "DISK"
    }
  }

  categories {
    name  = "Environment"
    value = "Staging"
  }

  categories {
    name  = "TIER-1"
    value = nutanix_category_value.WEB.id
  }
}

#DB-1 VM
resource "nutanix_virtual_machine" "vm-db" {
  name                 = "test-dou-vm-flow-DB-1"
  cluster_uuid         = local.cluster_uuid
  num_vcpus_per_socket = 1
  num_sockets          = 1
  memory_size_mib      = 186

  disk_list {
```

```
  nic_list {
    subnet_uuid = "c56b535c-8aff-4435-ae85-78e64a07f76d"
  }

  disk_list {
    data_source_reference = {
      kind = "image"
      uuid = nutanix_image.cirros-034-disk.id
    }

    device_properties {
      disk_address = {
        device_index = 0
        adapter_type = "SCSI"
      }
      device_type = "DISK"
    }
  }

  categories {
    name  = "Environment"
    value = "Staging"
  }

  categories {
    name  = "TIER-1"
    value = nutanix_category_value.DB.id
  }
}

//Create Application Network Policy.
resource "nutanix_network_security_rule" "TEST-TIER" {
  name        = "RULE-1-TIERS"
  description = "rule 1 tiers"

  app_rule_action = "APPLY"

  app_rule_inbound_allow_list {
    peer_specification_type = "FILTER"
    filter_type             = "CATEGORIES_MATCH_ALL"
    filter_kind_list        = ["vm"]

    filter_params {
      name   = "${nutanix_category_key.test-category-key.id}"
      values = ["${nutanix_category_value.WEB.id}"]
    }
  }



  app_rule_target_group_default_internal_policy = "DENY_ALL"

  app_rule_target_group_peer_specification_type = "FILTER"

  app_rule_target_group_filter_type = "CATEGORIES_MATCH_ALL"

  app_rule_target_group_filter_kind_list = ["vm"]

  app_rule_target_group_filter_params {
    name   = "${nutanix_category_key.test-category-key.id}"
    values = ["${nutanix_category_value.APP.id}"]
```

```
  }
  app_rule_target_group_filter_params {
    name   = "${nutanix_category_key.USER.id}"
    values = ["${nutanix_category_value.group.id}"]
  }

  app_rule_target_group_filter_params {
    name   = "AppType"
    values = ["Default"]
  }

  app_rule_outbound_allow_list {
    peer_specification_type = "FILTER"
    filter_type             = "CATEGORIES_MATCH_ALL"
    filter_kind_list        = ["vm"]

    filter_params {
      name   = "${nutanix_category_key.test-category-key.id}"
      values = ["${nutanix_category_value.DB.id}"]
    }
  }

  depends_on = [nutanix_virtual_machine.vm-app, nutanix_virtual_machine.vm-web, nutanix_virtual_machine.v
m-db]
}
```

Argument Reference

The following arguments are supported:

- `name` : - (Required) The name for the image.
- `categories` : - (Optional) Categories for the image.
- `project_reference` : - (Optional) The reference to a project.
- `owner_reference` : - (Optional) The reference to a user.
- `description` : - (Optional) A description for image.
- `quarantine_rule_action` : - (Optional) These rules are used for quarantining suspected VMs. Target group is a required attribute. Empty inbound_allow_list will not allow anything into target group. Empty outbound_allow_list will allow everything from target group.
- `quarantine_rule_outbound_allow_list` : - (Optional)
- `quarantine_rule_target_group_default_internal_policy` : - (Optional) - Default policy for communication within target group.
- `quarantine_rule_target_group_peer_specification_type` : - (Optional) - Way to identify the object for which rule is applied.
- `quarantine_rule_target_group_filter_kind_list` : - (Optional) - List of kinds associated with this filter.
- `quarantine_rule_target_group_filter_type` : - (Optional) - The type of the filter being used.

- `quarantine_rule_target_group_filter_params` : - (Optional) - A list of category key and list of values.
- `quarantine_rule_inbound_allow_list` : - (Optional)
- `app_rule_action` : - (Optional) - These rules govern what flows are allowed. Target group is a required attribute. Empty inbound_allow_list will not anything into target group. Empty outbound_allow_list will allow everything from target group.
- `app_rule_outbound_allow_list` : - (Optional)
- `app_rule_target_group_default_internal_policy` : - (Optional) - Default policy for communication within target group.
- `app_rule_target_group_peer_specification_type` : - (Optional) - Way to identify the object for which rule is applied.
- `app_rule_target_group_filter_kind_list` : - (Optional) - List of kinds associated with this filter.
- `app_rule_target_group_filter_type` : - (Optional) - The type of the filter being used.
- `app_rule_target_group_filter_params` : - (Optional) - A list of category key and list of values.
- `app_rule_inbound_allow_list` : - (Optional)
- `isolation_rule_action` : - (Optional) - These rules are used for environmental isolation.
- `app_rule_inbound_allow_list` : - (Optional)
- `isolation_rule_first_entity_filter_kind_list` : - (Optional) - List of kinds associated with this filter.
- `isolation_rule_first_entity_filter_type` : - (Optional) - The type of the filter being used.
- `isolation_rule_first_entity_filter_params` : - (Optional) - A list of category key and list of values.
- `isolation_rule_second_entity_filter_kind_list` : - (Optional) - List of kinds associated with this filter.
- `isolation_rule_second_entity_filter_type` : - (Optional) - The type of the filter being used.
- `isolation_rule_second_entity_filter_params` : - (Optional) - A list of category key and list of values.

Attributes Reference

The following attributes are exported:

- `metadata` : - The image kind metadata.
- `retrieval_uri_list` : - List of URIs where the raw image data can be accessed.
- `size_bytes` : - The size of the image in bytes.
- `state` : - The state of the Network Security Rule.
- `api_version` - The version of the API.

Metadata

The metadata attribute exports the following:

- `last_update_time` : - UTC date and time in RFC-3339 format when image was last updated.
- `UUID` : - image UUID.
- `creation_time` : - UTC date and time in RFC-3339 format when image was created.
- `spec_version` : - Version number of the latest spec.
- `spec_hash` : - Hash of the spec. This will be returned from server.
- `name` : - image name.

Reference

The `project_reference` , `owner_reference` , `availability_zone_reference` , `cluster_reference` , attributes supports the following:

- `kind` : - The kind name (Default value: project)(Required).
- `name` : - the name(Optional).
- `uuid` : - the UUID(Required).

See detailed information in Nutanix Image (https://nutanix.github.io/Automation/experimental/swagger-redoc-sandbox/#tag/network_security_rules/paths/~1network_security_rules~1{UUID}/put).

nutanix_subnet

Provides a resource to create a subnet based on the input parameters. A subnet is a block of IP addresses.

Example Usage

```
data "nutanix_clusters" "clusters" {
  metadata = {
    length = 2
  }
}

output "cluster" {
  value = data.nutanix_clusters.clusters.entities.0.metadata.uuid
}

resource "nutanix_subnet" "next-iac-managed" {
  # What cluster will this VLAN live on?
  cluster_uuid = "${data.nutanix_clusters.clusters.entities.0.metadata.uuid}"

  # General Information
  name        = "next-iac-managed-example"
  vlan_id     = 101
  subnet_type = "VLAN"

  # Managed L3 Networks
  # This bit is only needed if you intend to turn on IPAM
  prefix_length = 20

  default_gateway_ip = "10.5.80.1"
  subnet_ip          = "10.5.80.0"

  dhcp_domain_name_server_list = ["8.8.8.8", "4.2.2.2"]
  dhcp_domain_search_list      = ["nutanix.com", "eng.nutanix.com"]
}
```

Argument Reference

- `metadata` : - (Required) The subnet kind metadata.
- `availability_zone_reference` : - (Optional) The reference to a availability_zone.
- `cluster_uuid` : - (Required) The UUID of the cluster.
- `description` : - (Optional) A description for subnet.
- `name` : - (Optional) Subnet name (Readonly).
- `categories` : - (Optional) The categories of the resource.
- `owner_reference` : - (Optional) The reference to a user.

- `project_reference` : - (Optional) The reference to a project.
- `vswitch_name` : - (Optional).
- `subnet_type` : - (Optional).
- `default_gateway_ip` : - (Optional) Default gateway IP address.
- `prefix_length` : - (Optional).
- `subnet_ip` : - (Optional) Subnet IP address.
- `dhcp_server_address` : - (Optional) Host address.
- `dhcp_server_address_port` : - (Optional) Port Number.
- `dhcp_options` : - (Optional) Spec for defining DHCP options.
- `dhcp_domain_search_list` : - (Optional).
- `vlan_id` : - (Optional).
- `network_function_chain_reference` : - (Optional) The reference to a network_function_chain.

Attributes Reference

The following attributes are exported:

- `metadata` : - The vm kind metadata.
- `state` : - The state of the subnet.
- `api_version` - The version of the API.

Metadata

The metadata attribute exports the following:

- `last_update_time` : - UTC date and time in RFC-3339 format when subnet was last updated.
- `uuid` : - The subnet UUID.
- `creation_time` : - UTC date and time in RFC-3339 format when subnet was created.
- `spec_version` : - Version number of the latest spec.
- `spec_hash` : - Hash of the spec. This will be returned from server.
- `name` : - subnet name.

Categories

The categories attribute supports the following:

- `name` : - the key name.

- `value` : - value of the key.

Reference

The `project_reference` , `owner_reference` , `availability_zone_reference` , `network_function_chain_reference` , `subnet_reference` .

attributes supports the following:

- `kind` : - The kind name (Default value: project)(Required).
- `name` : - the name(Optional).
- `uuid` : - the UUID(Required).

Note: `subnet_reference` does not support the attribute `name`

See detailed information in Nutanix Subnet (http://developer.nutanix.com/reference/prism_central/v3/#definitions-subnet_resources).

nutanix_virtual_machine

Provides a Nutanix Virtual Machine resource to Create a virtual machine.

Example Usage

```
data "nutanix_clusters" "clusters" {}

resource "nutanix_virtual_machine" "vm1" {
  name = "test-dou"
  cluster_uuid = data.nutanix_clusters.clusters.entities.0.metadata.uuid

  categories {
        name   = "Environment"
    value  = "Staging"
    }


  num_vcpus_per_socket = 1
  num_sockets          = 1
  memory_size_mib      = 2048
}
```

Argument Reference

The following arguments are supported:

- `name` : - (Required) The name for the vm.
- `cluster_uuid` : - (Required) The UUID of the cluster.
- `categories` : - (Optional) Categories for the vm.
- `project_reference` : - (Optional) The reference to a project.
- `owner_reference` : - (Optional) The reference to a user.
- `availability_zone_reference` : - (Optional) The reference to a availability_zone.
- `description` : - (Optional) A description for vm.
- `num_vnuma_nodes` : - (Optional) Number of vNUMA nodes. 0 means vNUMA is disabled.
- `nic_list` : - (Optional) Spec NICs attached to the VM.
- `serial_port_list` : - (Optional) Serial Ports configured on the VM.
- `guest_os_id` : - (Optional) Guest OS Identifier. For ESX, refer to VMware documentation link (https://www.vmware.com/support/developer/converter-sdk/conv43_apireference/vim.vm.GuestOsDescriptor.GuestOsIdentifier.html) for the list of guest OS identifiers.
- `power_state` : - (Optional) The current or desired power state of the VM. (Options : ON , OFF)

- `nutanix_guest_tools` : - (Optional) Information regarding Nutanix Guest Tools.
- `ngt_credentials` : - (Ooptional) Credentials to login server.
- `ngt_enabled_capability_list` - (Optional) Application names that are enabled.
- `num_vcpus_per_socket` : - (Optional) Number of vCPUs per socket.
- `num_sockets` : - (Optional) Number of vCPU sockets.
- `gpu_list` : - (Optional) GPUs attached to the VM.
- `parent_referece` : - (Optional) Reference to an entity that the VM cloned from.
- `memory_size_mib` : - (Optional) Memory size in MiB.
- `boot_device_order_list` : - (Optional) Indicates the order of device types in which VM should try to boot from. If boot device order is not provided the system will decide appropriate boot device order.
- `boot_device_disk_address` : - (Optional) Address of disk to boot from.
- `boot_device_mac_address` : - (Optional) MAC address of nic to boot from.
- `hardware_clock_timezone` : - (Optional) VM's hardware clock timezone in IANA TZDB format (America/Los_Angeles).
- `guest_customization_cloud_init_user_data` : - (Optional) The contents of the user_data configuration for cloud-init. This can be formatted as YAML, JSON, or could be a shell script. The value must be base64 encoded.
- `guest_customization_cloud_init_meta_data` - (Optional) The contents of the meta_data configuration for cloud-init. This can be formatted as YAML or JSON. The value must be base64 encoded.
- `guest_customization_cloud_init_custom_key_values` : - (Optional) Generic key value pair used for custom attributes in cloud init.
- `guest_customization_is_overridable` : - (Optional) Flag to allow override of customization by deployer.
- `guest_customization_sysprep` : - (Optional) VM guests may be customized at boot time using one of several different methods. Currently, cloud-init w/ ConfigDriveV2 (for Linux VMs) and Sysprep (for Windows VMs) are supported. Only ONE OF sysprep or cloud_init should be provided. Note that guest customization can currently only be set during VM creation. Attempting to change it after creation will result in an error. Additional properties can be specified. For example - in the context of VM template creation if \"override_script\" is set to \"True\" then the deployer can upload their own custom script.
- `guest_customization_sysrep_custom_key_values` : - (Optional) Generic key value pair used for custom attributes in sysrep.
- `should_fail_on_script_failure` : - (Optional) Extra configs related to power state transition. Indicates whether to abort ngt shutdown/reboot if script fails.
- `enable_script_exec` : - (Optional) Extra configs related to power state transition. Indicates whether to execute set script before ngt shutdown/reboot.
- `power_state_mechanism` : - (Optional) Indicates the mechanism guiding the VM power state transition. Currently used for the transition to \"OFF\" state. Power state mechanism (ACPI/GUEST/HARD).
- `vga_console_enabled` : - (Optional) Indicates whether VGA console should be enabled or not.
- `disk_list` Disks attached to the VM.

Disk List

The disk_list attribute supports the following:

- `uuid` : - (Optional) The device ID which is used to uniquely identify this particular disk.
- `disk_size_bytes` - (Optional) Size of the disk in Bytes.
- `disk_size_mib` - Size of the disk in MiB. Must match the size specified in 'disk_size_bytes' - rounded up to the nearest MiB - when that field is present.
- `device_properties` - Properties to a device.
- `data_source_reference` - Reference to a data source.
- `volume_group_reference` - Reference to a volume group.

The disk_size (the disk size_mib and the disk_size_bytes attributes) is only honored by creating an empty disk. When you are creating from an image, the size is ignored and the disk becomes the size of the image from which it was cloned. In VM creation, you can't set either disk size_mib or disk_size_bytes when you set data_source_reference but, you can update the disk_size after creation (second apply).

Device Properties

The device_properties attribute supports the following.

- `device_type` : - A Disk type (default: DISK).
- `disk_address` : - Address of disk to boot from.

Sysprep

The guest_customization_sysprep attribute supports the following:

- `install_type` : - (Optional) Whether the guest will be freshly installed using this unattend configuration, or whether this unattend configuration will be applied to a pre-prepared image. Default is \"PREPARED\".
- `unattend_xml` : - (Optional) Generic key value pair used for custom attributes.

Disk Address

The boot_device_disk_address attribute supports the following:

- `device_index` : - (Optional) The index of the disk address.
- `adapter_type` : - (Optional) The adapter type of the disk address.

GPU List

The gpu_list attribute supports the following:

- `frame_buffer_size_mib` : - (ReadOnly) GPU frame buffer size in MiB.

- `vendor` : - (Optional) The vendor of the GPU.
- `uuid` : - (ReadOnly) UUID of the GPU.
- `name` : - (ReadOnly) Name of the GPU resource.
- `pci_address` - (ReadOnly) GPU {segment:bus:device:function} (sbdf) address if assigned.
- `fraction` - (ReadOnly) Fraction of the physical GPU assigned.
- `mode` : - (Optional) The mode of this GPU.
- `num_virtual_display_heads` : - (ReadOnly) Number of supported virtual display heads.
- `guest_driver_version` : - (ReadOnly) Last determined guest driver version.
- `device_id` : - (Computed) The device ID of the GPU.

Nutanix Guest Tools

The nutanix_guest_tools attribute supports the following:

- `state` : - (Optional) Nutanix Guest Tools is enabled or not.
- `ngt_state` : - (Optional) Nutanix Guest Tools is enabled or not.
- `iso_mount_state` : - (Optional) Desired mount state of Nutanix Guest Tools ISO.
- `version` : - (ReadOnly) Version of Nutanix Guest Tools installed on the VM.
- `available_version` : - (ReadOnly) Version of Nutanix Guest Tools available on the cluster.
- `guest_os_version` : - (ReadOnly) Version of the operating system on the VM.
- `vss_snapshot_capable` : - (ReadOnly) Whether the VM is configured to take VSS snapshots through NGT.
- `is_reachable` : - (ReadOnly) Communication from VM to CVM is active or not.
- `vm_mobility_drivers_installed` : - (ReadOnly) Whether VM mobility drivers are installed in the VM.

NIC List

The nic_list attribute supports the following:

- `nic_type` : - The type of this NIC. Defaults to NORMAL_NIC. (Options : NORMAL_NIC , DIRECT_NIC , NETWORK_FUNCTION_NIC).
- `uuid` : - The NIC's UUID, which is used to uniquely identify this particular NIC. This UUID may be used to refer to the NIC outside the context of the particular VM it is attached to.
- `model` : - The model of this NIC. (Options : VIRTIO , E1000).
- `network_function_nic_type` : - The type of this Network function NIC. Defaults to INGRESS. (Options : INGRESS , EGRESS , TAP).
- `mac_address` : - The MAC address for the adapter.

- `ip_endpoint_list` : - IP endpoints for the adapter. Currently, IPv4 addresses are supported.
- `network_function_chain_reference` : - The reference to a network_function_chain.
- `subnet_uuid` : - The reference to a subnet.
- `subnet_name` : - The name of the subnet reference to.
- `floating_ip` : - The Floating IP associated with the vnic. (Only in `nic_list_status`)

Serial Port List

The `serial_port_list` attribute supports the following:

- `index` : - Index of the serial port (int).
- `is_connected` : - Indicates whether the serial port connection is connected or not (`true` or `false`).

ip_endpoint_list

The following attributes are exported:

- `ip` : - Address string.
- `type` : - Address type. It can only be "ASSIGNED" in the spec. If no type is specified in the spec, the default type is set to "ASSIGNED". (Options : ASSIGNED , LEARNED)

Attributes Reference

The following attributes are exported:

- `metadata` : - The vm kind metadata.
- `api_version` - The version of the API.
- `state` : - The state of the vm.
- `cluster_name` : - The name of the cluster.
- `host_reference` : - Reference to a host.
- `hypervisor_type` : - The hypervisor type for the hypervisor the VM is hosted on.
- `nic_list_status` : - Status NICs attached to the VM.

Metadata

The metadata attribute exports the following:

- `last_update_time` : - UTC date and time in RFC-3339 format when vm was last updated.
- `uuid` : - vm UUID.

- `creation_time` : - UTC date and time in RFC-3339 format when vm was created.
- `spec_version` : - Version number of the latest spec.
- `spec_hash` : - Hash of the spec. This will be returned from server.
- `name` : - vm name.

Categories

The categories attribute supports the following:

- `name` : - the key name.
- `value` : - value of the key.

Reference

The `project_reference`, `owner_reference`, `availability_zone_reference`, `network_function_chain_reference`, `data_source_reference`, `volume_group_reference` attributes supports the following:

- `kind` : - The kind name (Default value: project)(Required).
- `name` : - the name(Optional).
- `uuid` : - the UUID(Required).

See detailed information in Nutanix Virtual Machine (http://developer.nutanix.com/reference/prism_central/v3/#vms).

nutanix_volume_group

Provides a Nutanix Volume Group resource to Create a volume_group.

Example Usage

```
resource "nutanix_volume_group" "test_volume" {
  name        = "Test Volume Group"
  description = "Test Volume Group Description"
}
```

Argument Reference

The following arguments are supported:

- `name` : - (Required) The name for the volume_group.
- `categories` : - (Optional) Categories for the volume_group.
- `project_reference` : - (Optional) The reference to a project.
- `owner_reference` : - (Optional) The reference to a user.
- `api_version` - (Optional) Version of the API.
- `description` : - (Optional) A description for volume_group.
- `flash_mode` : - (Optional) Flash Mode, if enabled all volume disks of the VG will be pinned to SSD tier.
- `file_system_type` : - (Optional) File system to be used for volume.
- `sharing_status` : - (Optiional) Whether the volume group can be shared across multiple iSCSI initiators.
- `attachment_list` : - (Optional) VMs attached to volume group.
- `disk_list` : - (Optional) Volume group disk specification.
- `iscsi_target_prefix` : - (Optional) iSCSI target prefix-name.

Disk List

The disk_list attribute supports the following:

- `vmdisk_UUID` : - (Optional) The UUID of this volume disk.
- `index` : - (Optional) Index of the volume disk in the group.
- `data_source_reference` : - (Optional) Reference to a kind
- `disk_size_mib` : - (Optional) Size of the disk in MiB.

- `storage_container_UUID` : - (Optional) Container UUID on which to create the disk.

Attachment List

The attachment_list attribute supports the following:

- `vm_reference` : - (Optional) Reference to a kind.
- `iscsi_initiator_name` : - (Optional) Name of the iSCSI initiator of the workload outside Nutanix cluster.

Attributes Reference

The following attributes are exported:

- `metadata` : - The volume_group kind metadata.
- `state` : - The state of the volume group.

Metadata

The metadata attribute exports the following:

- `last_update_time` : - UTC date and time in RFC-3339 format when volume_group was last updated.
- `UUID` : - volume_group UUID.
- `creation_time` : - UTC date and time in RFC-3339 format when volume_group was created.
- `spec_version` : - Version number of the latest spec.
- `spec_hash` : - Hash of the spec. This will be returned from server.
- `name` : - volume_group name.

Reference

The `project_reference` , `owner_reference` , `availability_zone_reference` , `cluster_reference` , attributes supports the following:

- `kind` : - The kind name (Default value: project)(Required).
- `name` : - the name(Optional).
- `UUID` : - the UUID(Required).

Note: `vm_reference` and `data_source_reference` don't support `name` argument.

See detailed information in Nutanix Volume Group (https://nutanix.github.io/Automation/experimental/swagger-redoc-sandbox/#tag/volume_group).