

## » **yandex\_\_compute\_\_disk**

Get information about a Yandex Compute disk. For more information, see the official documentation.

### » **Example Usage**

```
data "yandex_compute_disk" "my_disk" {
  disk_id = "some_disk_id"
}

resource "yandex_compute_instance" "default" {
  ...

  secondary_disk {
    disk_id = "${data.yandex_compute_disk.my_disk.id}"
  }
}
```

### » **Argument Reference**

The following arguments are supported:

- **disk\_id** - (Optional) The ID of a specific disk.
- **name** - (Optional) Name of the disk.

**NOTE:** One of **disk\_id** or **name** should be specified.

### » **Attributes Reference**

In addition to the arguments listed above, the following computed attributes are exported:

- **description** - Optional description of this disk.
- **folder\_id** - ID of the folder that the disk belongs to.
- **zone** - ID of the zone where the disk resides.
- **size** - Size of the disk, specified in Gb.
- **image\_id** - ID of the source image that was used to create this disk.
- **snapshot\_id** - Source snapshot that was used to create this disk.
- **type** - Type of the disk.
- **status** - Status of the disk.
- **labels** - Map of labels applied to this disk.
- **product\_ids** - License IDs that indicate which licenses are attached to this disk.

- `instance_ids` - IDs of instances to which this disk is attached.
- `created_at` - Disk creation timestamp.

## » `yandex__compute__image`

Get information about a Yandex Compute image. For more information, see the official documentation.

### » Example Usage

```
data "yandex_compute_image" "my_image" {
  family = "ubuntu-1804-lts"
}

resource "yandex_compute_instance" "default" {
  ...

  boot_disk {
    initialize_params {
      image_id = "${data.yandex_compute_image.my_image.id}"
    }
  }
}
```

### » Argument Reference

The following arguments are supported:

- `image_id` - (Optional) The ID of a specific image.
- `family` - (Optional) The family name of an image. Used to search the latest image in a family.
- `name` - (Optional) The name of the image.

**NOTE:** Either `image_id`, `family` or `name` must be specified.

- `folder_id` - (Optional) Folder that the resource belongs to. If a value is not provided, the default provider folder is used.

**NOTE:** If you specify `family` without `folder_id` then lookup takes place in the 'standard-images' folder.

## » Attributes Reference

In addition to the arguments listed above, the following computed attributes are exported:

- **description** - An optional description of this image.
- **family** - The OS family name of the image.
- **min\_disk\_size** - Minimum size of the disk which is created from this image.
- **size** - The size of the image, specified in Gb.
- **status** - The status of the image.
- **product\_ids** - License IDs that indicate which licenses are attached to this image.
- **os\_type** - Operating system type that the image contains.
- **labels** - A map of labels applied to this image.
- **created\_at** - Image creation timestamp.

## » `yandex_compute_instance`

Get information about a Yandex Compute instance. For more information, see the official documentation.

## » Example Usage

```
data "yandex_compute_instance" "my_instance" {
  instance_id = "some_instance_id"
}

output "instance_external_ip" {
  value = "${data.yandex_compute_instance.my_instance.network_interface.0.nat_ip_address}"
}
```

## » Argument Reference

The following arguments are supported:

- **instance\_id** - (Optional) The ID of a specific instance.
- **name** - (Optional) Name of the instance.

**NOTE:** One of `instance_id` or `name` should be specified.

## » Attributes Reference

- `description` - Description of the instance.
- `folder_id` - ID of the folder that the instance belongs to.
- `fqdn` - FQDN of the instance.
- `zone` - Availability zone where the instance resides.
- `labels` - A set of key/value label pairs to assign to the instance.
- `metadata` - Metadata key/value pairs to make available from within the instance.
- `platform_id` - Type of virtual machine to create. Default is 'standard-v1'.
- `status` - Status of the instance.
- `resources.0.memory` - Memory size allocated for the instance.
- `resources.0.cores` - Number of CPU cores allocated for the instance.
- `resources.0.core_fraction` - Baseline performance for a core, set as a percent.
- `boot_disk` - The boot disk for the instance. Structure is documented below.
- `network_interface` - The networks attached to the instance. Structure is documented below.
- `network_interface.0.ip_address` - An internal IP address of the instance, either manually or dynamically assigned.
- `network_interface.0.nat_ip_address` - An assigned external IP address if the instance has NAT enabled.
- `secondary_disk` - List of secondary disks attached to the instance. Structure is documented below.
- `scheduling_policy` - Scheduling policy configuration. The structure is documented below.
- `service_account_id` - ID of the service account authorized for this instance.
- `created_at` - Instance creation timestamp.

---

The `boot_disk` block supports:

- `auto_delete` - Whether the disk is auto-deleted when the instance is deleted. The default value is false.
- `device_name` - Name that can be used to access an attached disk under `/dev/disk/by-id/`.
- `mode` - Access to the disk resource. By default a disk is attached in `READ_WRITE` mode.
- `disk_id` - ID of the attached disk.
- `initialize_params` - Parameters used for creating a disk alongside the instance. The structure is documented below.

The `initialize_params` block supports:

- `name` - Name of the boot disk.

- **description** - Description of the boot disk.
- **size** - Size of the disk in GB.
- **type** - Disk type.
- **image\_id** - A disk image to initialize this disk from.
- **snapshot\_id** - A snapshot to initialize this disk from.

The **network\_interface** block supports:

- **index** - The index of the network interface, generated by the server.
- **mac\_address** - MAC address that is assigned to the network interface.
- **ip\_address** - The private IP address to assign to the instance. If empty, the address is automatically assigned from the specified subnet.
- **subnet\_id** - ID of the subnet to attach this interface to. The subnet must reside in the same zone where this instance was created.
- **nat** - Assigned for the instance's public address that is used to access the internet over NAT.
- **nat\_ip\_address** - Public IP address of the instance.
- **nat\_ip\_version** - IP version for the public address.

The **secondary\_disk** block supports:

- **auto\_delete** - Specifies whether the disk is auto-deleted when the instance is deleted.
- **device\_name** - This value can be used to reference the device from within the instance for mounting, resizing, and so on.
- **mode** - Access to the Disk resource. By default, a disk is attached in READ\_WRITE mode.
- **disk\_id** - ID of the disk that is attached to the instance.

The **scheduling\_policy** block supports:

- **preemptible** - (Optional) Specifies if the instance is preemptible. Defaults to false.

## » **yandex\_compute\_instance\_group**

Get information about a Yandex Compute instance group.

### » **Example Usage**

```
data "yandex_compute_instance_group" "my_group" {
  instance_group_id = "some_instance_group_id"
}
```

```
output "instance_external_ip" {
  value = "${data.yandex_compute_instance_group.my_group.network_interface.0.nat_ip_address}"
}
```

}

## » Argument Reference

The following arguments are supported:

- **instance\_group\_id** - (Required) The ID of a specific instance group.

## » Attributes Reference

- **name** - The name of the instance group.
- **description** - A description of the instance group.
- **folder\_id** - The ID of the folder that the instance group belongs to.
- **labels** - A set of key/value label pairs to assign to the instance group.
- **health\_check** - Health check specification.

The structure is documented below.

- **load\_balancer** - Load balancing specification.

The structure is documented below.

- **deploy\_policy** - The deployment policy of the instance group.

The structure is documented below.

- **allocation\_policy** - The allocation policy of the instance group by zone and region.

The structure is documented below.

- **instances** - A list of instances in the specified instance group.

The structure is documented below.

- **instance\_template** - The instance template that the instance group belongs to.

The structure is documented below.

- **service\_account\_id** - The ID of the service account authorized for this instance group.
- **scale\_policy** - The scaling policy of the instance group.

The structure is documented below.

- **load\_balancer\_state** - Information about which entities can be attached to this load balancer.

The structure is documented below.

- **created\_at** - The instance group creation timestamp.

---

The `load_balancer_state` block supports:

- `target_group_id` - The ID of the target group used for load balancing.
- `status_message` - The status message of the target group.

---

The `scale_policy` block supports:

- `fixed_scale` - The fixed scaling policy of the instance group.

The structure is documented below.

---

The `fixed_scale` block supports:

- `size` - The number of instances in the instance group.

---

The `instance_template` block supports:

- `description` - A description of the instance template.
- `platform_id` - The ID of the hardware platform configuration for the instance.
- `service_account_id` - The service account ID for the instance.
- `metadata` - The set of metadata **key:value** pairs assigned to this instance template. This includes custom metadata and predefined keys.
- `labels` - A map of labels applied to this instance.
- `resources.0.memory` - The memory size allocated to the instance.
- `resources.0.cores` - Number of CPU cores allocated to the instance.
- `resources.0.core_fraction` - Baseline core performance as a percent.
- `scheduling_policy` - The scheduling policy for the instance. The structure is documented below.
- `network_interface` - An array with the network interfaces that will be attached to the instance. The structure is documented below.
- `secondary_disk` - An array with the secondary disks that will be attached to the instance. The structure is documented below.
- `boot_disk` - The specifications for boot disks that will be attached to the instance.

The structure is documented below.

---

The `boot_disk` block supports:

- `device_name` - This value can be used to reference the device under `/dev/disk/by-id/`.
- `mode` - The access mode to the disk resource. By default a disk is attached in `READ_WRITE` mode.

- `initialize_params` - The parameters used for creating a disk alongside the instance.

The structure is documented below.

---

The `initialize_params` block supports:

- `description` - A description of the boot disk.
- `size` - The size of the disk in GB.
- `type` - The disk type.
- `image_id` - The disk image to initialize this disk from.
- `snapshot_id` - The snapshot to initialize this disk from.

---

The `secondary_disk` block supports:

- `device_name` - This value can be used to reference the device under `/dev/disk/by-id/`.
- `mode` - The access mode to the disk resource. By default a disk is attached in `READ_WRITE` mode.
- `initialize_params` - The parameters used for creating a disk alongside the instance. The structure is documented below.

---

The `initialize_params` block supports:

- `description` - A description of the boot disk.
- `size` - The size of the disk in GB.
- `type` - The disk type.
- `image_id` - The disk image to initialize this disk from.
- `snapshot_id` - The snapshot to initialize this disk from.

---

The `network_interface` block supports:

- `network_id` - The ID of the network.
- `subnet_ids` - The IDs of the subnets.
- `nat` - A public address that can be used to access the internet over NAT.

---

The `scheduling_policy` block supports:

- `preemptible` - Specifies if the instance is preemptible. Defaults to false.

---

The `instances` block supports:

- `instance_id` - The ID of the instance.



- **name** - The name of the managed instance.
- **fqdn** - The Fully Qualified Domain Name.
- **status** - The status of the instance.
- **status\_message** - The status message of the instance.
- **zone\_id** - The ID of the availability zone where the instance resides.
- **network\_interface** - An array with the network interfaces attached to the managed instance.

The structure is documented below.

---

The **network\_interface** block supports:

- **index** - The index of the network interface as generated by the server.
- **mac\_address** - The MAC address assigned to the network interface.
- **ip\_address** - The private IP address to assign to the instance. If empty, the address is automatically assigned from the specified subnet.
- **subnet\_id** - The ID of the subnet to attach this interface to. The subnet must reside in the same zone where this instance was created.
- **nat** - The instance's public address for accessing the internet over NAT.
- **nat\_ip\_address** - The public IP address of the instance.
- **nat\_ip\_version** - The IP version for the public address.

---

The **allocation\_policy** block supports:

- **zones** - A list of availability zones.

---

The **deploy\_policy** block supports:

- **max\_unavailable** - The maximum number of running instances that can be taken offline (stopped or deleted) at the same time during the update process.
- **max\_expansion** - The maximum number of instances that can be temporarily allocated above the group's target size during the update process.
- **max\_deleting** - The maximum number of instances that can be deleted at the same time.
- **max\_creating** - The maximum number of instances that can be created at the same time.
- **startup\_duration** - The amount of time in seconds to allow for an instance to start.

Instance will be considered up and running (and start receiving traffic) only after the **startup\_duration** has elapsed and all health checks are passed.

---

The **load\_balancer** block supports:

- `target_group_name` - The name of the target group.
  - `target_group_description` - A description of the target group.
  - `target_group_labels` - A set of key/value label pairs.
  - `target_group_id` - The ID of the target group.
  - `status_message` - The status message of the target group.
- 

The `health_check` block supports:

- `interval` - The interval between health checks in seconds.
- `timeout` - Timeout for the managed instance to return a response for the health check in seconds.
- `healthy_threshold` - The number of successful health checks before the managed instance is declared healthy.
- `unhealthy_threshold` - The number of failed health checks before the managed instance is declared unhealthy.
- `tcp_options` - TCP check options.

The structure is documented below.

- `http_options` - HTTP check options.

The structure is documented below.

---

The `http_options` block supports:

- `port` - The port used for HTTP health checks.
  - `path` - The URL path used for health check requests.
- 

The `tcp_options` block supports:

- `port` - The port to use for TCP health checks.

## » `yandex_compute_snapshot`

Get information about a Yandex Compute snapshot. For more information, see the official documentation.

### » Example Usage

```
data "yandex_compute_snapshot" "my_snapshot" {
  snapshot_id = "some_snapshot_id"
}
```

```
resource "yandex_compute_instance" "default" {
  ...

  boot_disk {
    initialize_params {
      snapshot_id = "${data.yandex_compute_snapshot.my_snapshot.id}"
    }
  }
}
```

## » Argument Reference

The following arguments are supported:

- `snapshot_id` - (Optional) The ID of a specific snapshot.
- `name` - (Optional) The name of the snapshot.

**NOTE:** One of `snapshot_id` or `name` should be specified.

## » Attributes Reference

In addition to the arguments listed above, the following computed attributes are exported:

- `description` - An optional description of this snapshot.
- `folder_id` - ID of the folder that the snapshot belongs to.
- `storage_size` - The size of the snapshot, specified in Gb.
- `status` - The status of the snapshot.
- `disk_size` - Minimum required size of the disk which is created from this snapshot.
- `source_disk_id` - ID of the source disk.
- `labels` - A map of labels applied to this snapshot.
- `product_ids` - License IDs that indicate which licenses are attached to this snapshot.
- `created_at` - Snapshot creation timestamp.

## » `yandex__iam__policy`

Generates an IAM policy document that may be referenced by and applied to other Yandex.Cloud Platform resources, such as the `yandex_resourcemanager_folder` resource.

```
data "yandex_iam_policy" "admin" {
  binding {
```

```

    role = "admin"

    members = [
        "userAccount:user_id_1"
    ]
}

binding {
    role = "viewer"

    members = [
        "userAccount:user_id_2"
    ]
}
}

```

This data source is used to define IAM policies to apply to other resources. Currently, defining a policy through a data source and referencing that policy from another resource is the only way to apply an IAM policy to a resource.

## » Argument Reference

The following arguments are supported:

- **binding** (Required) - A nested configuration block (described below) that defines a binding to be included in the policy document. Multiple **binding** arguments are supported.

Each policy document configuration must have one or more **binding** blocks. Each block accepts the following arguments:

- **role** (Required) - The role/permission that will be granted to the members. See the IAM Roles documentation for a complete list of roles.
- **members** (Required) - An array of identities that will be granted the privilege in the **role**. Each entry can have one of the following values:
  - **userAccount:{user\_id}**: A unique user ID that represents a specific Yandex account.
  - **serviceAccount:{service\_account\_id}**: A unique service account ID.

## » Attributes Reference

The following attribute is exported:

- **policy\_data** - The above bindings serialized in a format suitable for referencing from a resource that supports IAM.

## » **yandex\_\_iam\_\_role**

Generates an IAM role document that may be referenced by and applied to other Yandex.Cloud Platform resources, such as the **yandex\_resourcemanager\_folder** resource. For more information, see the official documentation.

```
data "yandex_iam_role" "admin" {
  binding {
    role = "admin"

    members = [
      "userAccount:user_id_1"
    ]
  }
}
```

This data source is used to define IAM roles in order to apply them to other resources. Currently, defining a role through a data source and referencing that role from another resource is the only way to apply an IAM role to a resource.

## » **Argument Reference**

The following arguments are supported:

- **binding** (Required) - A nested configuration block (described below) that defines a binding to be included in the policy document. Multiple **binding** arguments are supported.

Each role document configuration must have one or more **binding** blocks. Each block accepts the following arguments:

- **role** (Required) - The role/permission that will be granted to the members. See the IAM Roles documentation for a complete list of roles.
- **members** (Required) - An array of identities that will be granted the privilege in the **role**. Each entry can have one of the following values:
  - **userAccount:{user\_id}**: A unique user ID that represents a specific Yandex account.
  - **serviceAccount:{service\_account\_id}**: A unique service account ID.

## » Attributes Reference

The following attribute is exported:

- **role\_data** - The above bindings serialized in a format suitable for referencing from a resource that supports IAM.

## » `yandex_iam_service_account`

Get information about a Yandex IAM service account. For more information about accounts, see Yandex.Cloud IAM users.

```
data "yandex_iam_service_account" "builder" {
  service_account_id = "sa_id"
}
```

## » Argument reference

- **name** - Name of the service account. Can be updated without creating a new resource.
- **description** - Description of the service account.
- **folder\_id** - ID of the folder that the service account will be created in. If omitted, the provider folder configuration is used by default.

## » `yandex_iam_user`

Get information about a Yandex IAM user account. For more information about accounts, see Yandex.Cloud IAM users

```
data "yandex_iam_user" "admin" {
  login = "my-yandex-login"
}
```

This data source is used to define IAM Users that can be used by other resources.

## » Argument Reference

The following arguments are supported:

- **login** (Optional) - Login name used to sign in to Yandex Passport.
- **user\_id** (Optional) - User ID used to manage IAM access bindings.

**NOTE:** Either `login` or `user_id` must be specified.

## » Attributes Reference

The following attribute is exported:

- `user_id` - ID of IAM user account.
- `login` - Login name of IAM user account.
- `default_email` - Email address of user account.

## » `yandex_resourcemanager_cloud`

Use this data source to get cloud details. For more information, see [Cloud](#).

## » Example Usage

```
data "yandex_resourcemanager_cloud" "foo" {
  name = "foo-cloud"
}

output "cloud_create_timestamp" {
  value = "${data.yandex_resourcemanager_cloud.foo.created_at}"
}
```

## » Argument Reference

The following arguments are supported:

- `cloud_id` - (Optional) ID of the cloud.
- `name` - (Optional) Name of the cloud.

**NOTE:** Either `cloud_id` or `name` must be specified.

## » Attributes Reference

The following attributes are returned:

- `name` - Name of the cloud.
- `description` - Description of the cloud.
- `created_at` - Cloud creation timestamp.

## » `yandex_resourcemanager_folder`

Use this data source to get information about a Yandex Resource Manager Folder. For more information, see the official documentation.

```
# Get folder by ID
data "yandex_resourcemanager_folder" "my_folder_1" {
  folder_id = "folder_id_number_1"
}

# Get folder by name in specific cloud
data "yandex_resourcemanager_folder" "my_folder_2" {
  name      = "folder_name"
  cloud_id = "some_cloud_id"
}

output "my_folder_1_name" {
  value = "${data.yandex_resourcemanager_folder.my_folder_1.name}"
}

output "my_folder_2_cloud_id" {
  value = "${data.yandex_resourcemanager_folder.my_folder_2.cloud_id}"
}
```

## » Argument Reference

The following arguments are supported:

- `folder_id` (Optional) - ID of the folder.
- `name` (Optional) - Name of the folder.

**NOTE:** Either `folder_id` or `name` must be specified.

- `cloud_id` - (Optional) Cloud that the resource belongs to. If a value is not provided, the default provider cloud is used.

## » Attributes Reference

The following attributes are exported:

- `description` - Description of the folder.
- `cloud_id` - ID of the cloud that contains the folder.
- `status` - Current status of the folder.
- `labels` - A map of labels applied to this folder.
- `created_at` - Folder creation timestamp.



## » **yandex\_\_vpc\_\_network**

Get information about a Yandex VPC network. For more information, see [Yandex.Cloud VPC](#).

```
data "yandex_vpc_network" "admin" {
  network_id = "my-network-id"
}
```

This data source is used to define VPC Networks that can be used by other resources.

### » **Argument Reference**

The following arguments are supported:

- **network\_id** (Optional) - ID of the network.
- **name** (Optional) - Name of the network.

**NOTE:** One of **network\_id** or **name** should be specified.

### » **Attributes Reference**

The following attribute is exported:

- **description** - Description of the network.
- **folder\_id** - ID of the folder that the resource belongs to.
- **labels** - Labels assigned to this network.
- **created\_at** - Creation timestamp of this network.

## » **yandex\_\_vpc\_\_route\_\_table**

Get information about a Yandex VPC route table. For more information, see [Yandex.Cloud VPC](#).

```
data "yandex_vpc_route_table" "this" {
  route_table_id = "my-rt-id"
}
```

This data source is used to define VPC Route Table that can be used by other resources.

## » Argument Reference

The following arguments are supported:

- `route_table_id` (Optional) - Route table ID.
- `name` - (Optional) - Name of the route table.

**NOTE:** One of `route_table_id` or `name` should be specified.

## » Attributes Reference

The following attribute is exported:

- `description` - Description of the route table.
- `folder_id` - ID of the folder that the resource belongs to.
- `network_id` - ID of the network this route table belongs to.
- `labels` - Labels to assign to this route table.
- `static_route` - List of static route records of the route table. Structure is documented below.
- `created_at` - Creation timestamp of this route table.

The `static_route` block supports:

- `destination_prefix` - Route prefix in CIDR notation.
- `next_hop_address` - Address of the next hop.

## » `yandex_vpc_subnet`

Get information about a Yandex VPC subnet. For more information, see [Yandex.Cloud VPC](#).

```
data "yandex_vpc_subnet" "admin" {
  subnet_id = "my-subnet-id"
}
```

This data source is used to define VPC Subnets that can be used by other resources.

## » Argument Reference

The following arguments are supported:

- `subnet_id` (Optional) - Subnet ID.
- `name` - (Optional) - Name of the subnet.

**NOTE:** One of `subnet_id` or `name` should be specified.

## » Attributes Reference

The following attribute is exported:

- **description** - Description of the subnet.
- **folder\_id** - ID of the folder that the resource belongs to.
- **network\_id** - ID of the network this subnet belongs to.
- **labels** - Labels to assign to this subnet.
- **zone** - Name of the availability zone for this subnet.
- **route\_table\_id** - ID of the route table to assign to this subnet.
- **v4\_cidr\_blocks** - The blocks of internal IPv4 addresses owned by this subnet.
- **v6\_cidr\_blocks** - The blocks of internal IPv6 addresses owned by this subnet.
- **created\_at** - Creation timestamp of this subnet.

**Note:** **v6\_cidr\_blocks** attribute is currently not supported. It will be available in the future.

## » yandex\_lb\_network\_load\_balancer

Get information about a Yandex Load Balancer network load balancer. For more information, see Yandex.Cloud Network Load Balancer.

```
data "yandex_lb_network_load_balancer" "foo" {
  network_load_balancer_id = "my-network-load-balancer"
}
```

This data source is used to define Load Balancer Network Load Balancers that can be used by other resources.

## » Argument Reference

The following arguments are supported:

- **network\_load\_balancer\_id** (Optional) - Network load balancer ID.
- **name** - (Optional) - Name of the network load balancer.

**NOTE:** One of **network\_load\_balancer\_id** or **name** should be specified.

## » Attributes Reference

The following attribute is exported:

- **name** - Name of the network load balancer.

- `description` - Description of the network load balancer.
- `folder_id` - ID of the folder that the resource belongs to.
- `labels` - Labels to assign to this network load balancer.
- `region_id` - ID of the region where the network load balancer resides.
- `type` - Type of the network load balancer.
- `attached_target_group` - An attached target group is a group of targets that is attached to a load balancer. Structure is documented below.
- `listener` - Listener specification that will be used by a network load balancer. Structure is documented below.
- `created_at` - Creation timestamp of this network load balancer.

---

The `attached_target_group` block supports:

- `target_group_id` - ID of the target group that attached to the network load balancer.
- `healthcheck.0.name` - Name of the health check.
- `healthcheck.0.interval` - The interval between health checks.
- `healthcheck.0.timeout` - Timeout for a target to return a response for the health check.
- `healthcheck.0.unhealthy_threshold` - Number of failed health checks before changing the status to UNHEALTHY.
- `healthcheck.0.healthy_threshold` - Number of successful health checks required in order to set the HEALTHY status for the target.
- `healthcheck.0.tcp_options.0.port` - Port to use for TCP health checks.
- `healthcheck.0.http_options.0.port` - Port to use for HTTP health checks.
- `healthcheck.0.http_options.0.path` - URL path to use for HTTP health checks.

The `listener` block supports:

- `name` - Name of the listener.
- `port` - Port for incoming traffic.
- `protocol` - Protocol for incoming traffic.
- `target_port` - Port of a target.
- `external_address_spec.0.address` - Public IP address of a listener.
- `external_address_spec.0.ip_version` - IP version of the addresses.

## » `yandex_lb_target_group`

Get information about a Yandex Load Balancer target group. For more information, see [Yandex.Cloud Load Balancer](#).

```
data "yandex_lb_target_group" "foo" {
```

```
    target_group_id = "my-target-group-id"
}
```

This data source is used to define Load Balancer Target Groups that can be used by other resources.

## » Argument Reference

The following arguments are supported:

- `target_group_id` (Optional) - Target Group ID.
- `name` - (Optional) - Name of the Target Group.

**NOTE:** One of `target_group_id` or `name` should be specified.

## » Attributes Reference

The following attribute is exported:

- `name` - Name of the target group.
- `description` - Description of the target group.
- `folder_id` - ID of the folder that the resource belongs to.
- `labels` - Labels to assign to this target group.
- `target.0.address` - IP address of the target.
- `target.0.subnet_id` - ID of the subnet that targets are connected to.
- `created_at` - Creation timestamp of this target group.

## » `yandex__compute__disk`

Persistent disks are used for data storage and function similarly to physical hard and solid state drives.

A disk can be attached or detached from the virtual machine and can be located locally. A disk can be moved between virtual machines within the same availability zone. Each disk can be attached to only one virtual machine at a time.

For more information about disks in Yandex.Cloud, see:

- Documentation
- How-to Guides
  - Attach and detach a disk
  - Backup operation

## » Example Usage

```
resource "yandex_compute_disk" "default" {
  name      = "disk"
  type      = "network-nvme"
  zone      = "ru-central1-a"
  image_id  = "ubuntu-16.04-v20180727"

  labels = {
    environment = "test"
  }
}
```

## » Argument Reference

The following arguments are supported:

- **name** - (Optional) Name of the disk. Provide this property when you create a resource.
- **description** - (Optional) Description of the disk. Provide this property when you create a resource.
- **folder\_id** - (Optional) The ID of the folder that the disk belongs to. If it is not provided, the default provider folder is used.
- **labels** - (Optional) Labels to assign to this disk. A list of key/value pairs.
- **zone** - (Optional) Availability zone where the disk will reside.
- **size** - (Optional) Size of the persistent disk, specified in GB. You can specify this field when creating a persistent disk using the **image\_id** or **snapshot\_id** parameter, or specify it alone to create an empty persistent disk. If you specify this field along with **image\_id** or **snapshot\_id**, the size value must not be less than the size of the source image or the size of the snapshot.
- **type** - (Optional) Type of disk to create. Provide this when creating a disk. One of **network-hdd** (default) or **network-nvme**.
- **image\_id** - (Optional) The source image to use for disk creation.
- **snapshot\_id** - (Optional) The source snapshot to use for disk creation.

**NOTE:** Only one of **image\_id** or **snapshot\_id** can be specified.

## » Attributes Reference

In addition to the arguments listed above, the following computed attributes are exported:

- `status` - The status of the disk.
- `created_at` - Creation timestamp of the disk.

## » Timeouts

This resource provides the following configuration options for timeouts:

- `create` - Default is 5 minutes.
- `update` - Default is 5 minutes.
- `delete` - Default is 5 minutes.

## » Import

A disk can be imported using any of these accepted formats:

```
$ terraform import yandex_compute_disk.default disk_id
```

## » `yandex__compute__image`

Creates a virtual machine image resource for the Yandex Compute Cloud service from an existing tarball. For more information, see the official documentation.

## » Example Usage

```
resource "yandex_compute_image" "foo-image" {
  name      = "my-custom-image"
  source_url = "https://storage.yandexcloud.net/lucky-images/kube-it.img"
}

resource "yandex_compute_instance" "vm" {
  name = "vm-from-custom-image"

  # ...

  boot_disk {
    initialize_params {
      image_id = "${yandex_compute_image.foo-image.id}"
    }
  }
}
```

```

    }
  }
}

```

## » Argument Reference

The following arguments are supported:

- **name** - (Optional) Name of the disk.
- **description** - (Optional) An optional description of the image. Provide this property when you create a resource.
- **folder\_id** - (Optional) The ID of the folder that the resource belongs to. If it is not provided, the default provider folder is used.
- **labels** - (Optional) A set of key/value label pairs to assign to the image.
- **family** - (Optional) The name of the image family to which this image belongs.
- **min\_disk\_size** - (Optional) Minimum size in GB of the disk that will be created from this image.
- **os\_type** - (Optional) Operating system type that is contained in the image. Possible values: "LINUX", "WINDOWS".
- **source\_family** - (Optional) The name of the family to use as the source of the new image. The ID of the latest image is taken from the "standard-images" folder. Changing the family forces a new resource to be created.
- **source\_image** - (Optional) The ID of an existing image to use as the source of the image. Changing this ID forces a new resource to be created.
- **source\_snapshot** - (Optional) The ID of a snapshot to use as the source of the image. Changing this ID forces a new resource to be created.
- **source\_disk** - (Optional) The ID of a disk to use as the source of the image. Changing this ID forces a new resource to be created.
- **source\_url** - (Optional) The URL to use as the source of the image. Changing this URL forces a new resource to be created.
- **product\_ids** - (Optional) License IDs that indicate which licenses are attached to this image.

**NOTE:** One of `source_family`, `source_image`, `source_snapshot`, `source_disk` or `source_url` must be specified.



## » Attributes Reference

In addition to the arguments listed above, the following computed attributes are exported:

- `size` - The size of the image, specified in GB.
- `status` - The status of the image.
- `created_at` - Creation timestamp of the image.

## » Timeouts

`yandex_compute_image` provides the following configuration options for timeouts:

- `create` - Default 5 minutes
- `update` - Default 5 minutes
- `delete` - Default 5 minutes

## » Import

A VM image can be imported using the `id` of the resource, e.g.

```
$ terraform import yandex_compute_image.web-image image_id
```

## » `yandex__compute__instance`

A VM instance resource. For more information, see the official documentation.

## » Example Usage

```
resource "yandex_compute_instance" "default" {
  name          = "test"
  platform_id   = "standard-v1"
  zone          = "ru-central1-a"

  resources {
    cores   = 2
    memory  = 4
  }

  boot_disk {
    initialize_params {
      image_id = "image_id"
    }
  }
}
```

```

    }
  }

  network_interface {
    subnet_id = "${yandex_vpc_subnet.foo.id}"
  }

  metadata = {
    foo      = "bar"
    ssh-keys = "ubuntu:${file("~/ssh/id_rsa.pub")}"
  }
}

resource "yandex_vpc_network" "foo" {}

resource "yandex_vpc_subnet" "foo" {
  zone      = "ru-central1-a"
  network_id = "${yandex_vpc_network.foo.id}"
}

```

## » Argument Reference

The following arguments are supported:

- **resources** - (Required) Compute resources that are allocated for the instance. The structure is documented below.
- **boot\_disk** - (Required) The boot disk for the instance. The structure is documented below.
- **network\_interface** - (Required) Networks to attach to the instance. This can be specified multiple times. The structure is documented below.

- 
- **name** - (Optional) Resource name.
  - **description** - (Optional) Description of the instance.
  - **folder\_id** - (Optional) The ID of the folder that the resource belongs to. If it is not provided, the default provider folder is used.
  - **labels** - (Optional) A set of key/value label pairs to assign to the instance.
  - **zone** - (Optional) The availability zone where the virtual machine will be created. If it is not provided, the default provider folder is used.
  - **hostname** - (Optional) Host name for the instance. This field is used to generate the instance `fqdn` value. The host name must be unique within the network and region. If not specified, the host name will be equal to `id`

of the instance and `fqdn` will be `<id>.auto.internal`. Otherwise FQDN will be `<hostname>.<region_id>.internal`.

- **metadata** - (Optional) Metadata key/value pairs to make available from within the instance.
- **platform\_id** - (Optional) The type of virtual machine to create. The default is 'standard-v1'.
- **secondary\_disk** - (Optional) A list of disks to attach to the instance. The structure is documented below. **Note:** The **allow\_stopping\_for\_update** property must be set to true in order to update this structure.
- **scheduling\_policy** - (Optional) Scheduling policy configuration. The structure is documented below.
- **service\_account\_id** - (Optional) ID of the service account authorized for this instance.
- **allow\_stopping\_for\_update** - (Optional) If true, allows Terraform to stop the instance in order to update its properties. If you try to update a property that requires stopping the instance without setting this field, the update will fail.

---

The **resources** block supports:

- **cores** - (Required) CPU cores for the instance.
- **memory** - (Required) Memory size in GB.
- **core\_fraction** - (Optional) If provided, specifies baseline performance for a core as a percent.

The **boot\_disk** block supports:

- **auto\_delete** - (Optional) Defines whether the disk will be auto-deleted when the instance is deleted. The default value is **True**.
- **device\_name** - (Optional) Name that can be used to access an attached disk.
- **mode** - (Optional) Type of access to the disk resource. By default, a disk is attached in **READ\_WRITE** mode.
- **disk\_id** - (Optional) The ID of the existing disk (such as those managed by **yandex\_compute\_disk**) to attach as a boot disk.
- **initialize\_params** - (Optional) Parameters for a new disk that will be created alongside the new instance. Either **initialize\_params** or **disk\_id** must be set. The structure is documented below.

**NOTE:** Either **initialize\_params** or **disk\_id** must be specified.

The `initialize_params` block supports:

- `name` - (Optional) Name of the boot disk.
- `description` - (Optional) Description of the boot disk.
- `size` - (Optional) Size of the disk in GB.
- `type` - (Optional) Disk type.
- `image_id` - (Optional) A disk image to initialize this disk from.
- `snapshot_id` - (Optional) A snapshot to initialize this disk from.

**NOTE:** Either `image_id` or `snapshot_id` must be specified.

The `network_interface` block supports:

- `subnet_id` - (Required) ID of the subnet to attach this interface to. The subnet must exist in the same zone where this instance will be created.
- `ip_address` - (Optional) The private IP address to assign to the instance. If empty, the address will be automatically assigned from the specified subnet.
- `ipv6` - (Optional) If true, allocate an IPv6 address for the interface. The address will be automatically assigned from the specified subnet.
- `ipv6_address` - (Optional) The private IPv6 address to assign to the instance.
- `nat` - (Optional) Provide a public address, for instance, to access the internet over NAT.

The `secondary_disk` block supports:

- `disk_id` - (Required) ID of the disk that is attached to the instance.
- `auto_delete` - (Optional) Whether the disk is auto-deleted when the instance is deleted. The default value is false.
- `device_name` - (Optional) Name that can be used to access an attached disk under `/dev/disk/by-id/`.
- `mode` - (Optional) Type of access to the disk resource. By default, a disk is attached in `READ_WRITE` mode.

The `scheduling_policy` block supports:

- `preemptible` - (Optional) Specifies if the instance is preemptible. Defaults to false.

## » Attributes Reference

In addition to the arguments listed above, the following computed attributes are exported:

- `fqdn` - The fully qualified DNS name of this instance.
- `network_interface.0.address` - The internal IP address of the instance.
- `network_interface.0.nat_ip_address` - The external IP address of the instance.
- `status` - The status of this instance.
- `created_at` - Creation timestamp of the instance.

## » Import

Instances can be imported using the ID of an instance, e.g.

```
$ terraform import yandex_compute_instance.default instance_id
```

## » `yandex_compute_instance_group`

An Instance group resource. For more information, see the official documentation.

## » Example Usage

```
resource "yandex_compute_instance_group" "group1" {
  name          = "test-ig"
  folder_id     = "${data.yandex_resourcemanager_folder.test_folder.id}"
  service_account_id = "${yandex_iam_service_account.test_account.id}"
  instance_template {
    platform_id = "standard-v1"
    resources {
      memory = 1
      cores  = 1
    }
  }
  boot_disk {
    mode = "READ_WRITE"
    initialize_params {
      image_id = "${data.yandex_compute_image.ubuntu.id}"
      size     = 4
    }
  }
}
```

```

    }
    network_interface {
        network_id = "${yandex_vpc_network.my-inst-group-network.id}"
        subnet_ids = ["${yandex_vpc_subnet.my-inst-group-subnet.id}"]
    }
    labels = {
        label1 = "label1-value"
        label2 = "label2-value"
    }
    metadata = {
        foo      = "bar"
        ssh-keys = "ubuntu:${file("~/ssh/id_rsa.pub")}"
    }
}

scale_policy {
    fixed_scale {
        size = 3
    }
}

allocation_policy {
    zones = ["ru-central1-a"]
}

deploy_policy {
    max_unavailable = 2
    max_creating     = 2
    max_expansion    = 2
    max_deleting     = 2
}
}

```

## » Argument Reference

The following arguments are supported:

- **folder\_id** - (Required) The ID of the folder that the resources belong to.
- **scale\_policy** - (Required) The scaling policy of the instance group. The structure is documented below.
- **deploy\_policy** - (Required) The deployment policy of the instance group. The structure is documented below.
- **service\_account\_id** - (Required) The ID of the service account authorized for this instance group.

- **instance\_template** - (Required) The template for creating new instances. The structure is documented below.
- **allocation\_policy** - (Required) The allocation policy of the instance group by zone and region. The structure is documented below.

- 
- **name** - (Optional) The name of the instance group.
  - **health\_check** - (Optional) Health check specifications. The structure is documented below.
  - **load\_balancer** - (Optional) Load balancing specifications. The structure is documented below.
  - **description** - (Optional) A description of the instance group.
  - **labels** - (Optional) A set of key/value label pairs to assign to the instance group.
- 

The **load\_balancer** block supports:

- **target\_group\_name** - (Optional) The name of the target group.
  - **target\_group\_description** - (Optional) A description of the target group.
  - **target\_group\_labels** - (Optional) A set of key/value label pairs.
  - **target\_group\_id** - The ID of the target group.
  - **status\_message** - The status message of the target group.
- 

The **health\_check** block supports:

- **interval** - (Optional) The interval to wait between health checks in seconds.
- **timeout** - (Optional) The length of time to wait for a response before the health check times out in seconds.
- **healthy\_threshold** - (Optional) The number of successful health checks before the managed instance is declared healthy.
- **unhealthy\_threshold** - (Optional) The number of failed health checks before the managed instance is declared unhealthy.
- **tcp\_options** - (Optional) TCP check options. The structure is documented below.
- **http\_options** - (Optional) HTTP check options. The structure is documented below.

---

The `http_options` block supports:

- `port` - (Required) The port used for HTTP health checks.
  - `path` - (Required) The URL path used for health check requests.
- 

The `tcp_options` block supports:

- `port` - (Required) The port used for TCP health checks.
- 

The `allocation_policy` block supports:

- `zones` - (Required) A list of availability zones.
- 

The `instance_template` block supports:

- `boot_disk` - (Required) Boot disk specifications for the instance. The structure is documented below.
  - `resources` - (Required) Compute resource specifications for the instance. The structure is documented below.
  - `network_interface` - (Required) Network specifications for the instance. This can be used multiple times for adding multiple interfaces. The structure is documented below.
- 
- `scheduling_policy` - (Optional) The scheduling policy configuration. The structure is documented below.
  - `description` - (Optional) A description of the instance.
  - `metadata` - (Optional) A set of metadata key/value pairs to make available from within the instance.
  - `labels` - (Optional) A set of key/value label pairs to assign to the instance.
  - `platform_id` - (Optional) The ID of the hardware platform configuration for the instance. The default is 'standard-v1'.
  - `secondary_disk` - (Optional) A list of disks to attach to the instance. The structure is documented below.
  - `service_account_id` - (Optional) The ID of the service account authorized for this instance.
- 

The `secondary_disk` block supports:



- **mode** - (Required) The access mode to the disk resource. By default a disk is attached in `READ_WRITE` mode.
- **initialize\_params** - (Required) Parameters used for creating a disk alongside the instance. The structure is documented below.

- 
- **device\_name** - (Optional) This value can be used to reference the device under `/dev/disk/by-id/`.
- 

The **initialize\_params** block supports:

- **description** - (Optional) A description of the boot disk.
- **size** - (Optional) The size of the disk in GB.
- **type** - (Optional) The disk type.
- **image\_id** - (Optional) The disk image to initialize this disk from.
- **snapshot\_id** - (Optional) The snapshot to initialize this disk from.

**NOTE:** **image\_id** or **snapshot\_id** must be specified.

---

The **scheduling\_policy** block supports:

- **preemptible** - (Optional) Specifies if the instance is preemptible. Defaults to false.
- 

The **network\_interface** block supports:

- **network\_id** - (Optional) The ID of the network.
  - **subnet\_ids** - (Optional) The ID of the subnets to attach this interface to.
  - **nat** - (Optional) A public address that can be used to access the internet over NAT.
- 

The **resources** block supports:

- **memory** - (Required) The memory size in GB.
  - **cores** - (Required) The number of CPU cores for the instance.
- 
- **core\_fraction** - (Optional) If provided, specifies baseline core performance as a percent.

---

The `boot_disk` block supports:

- `mode` - (Required) The access mode to the disk resource. By default a disk is attached in `READ_WRITE` mode.
- `initialize_params` - (Required) Parameters for creating a disk alongside the instance. The structure is documented below.

- 
- `device_name` - (Optional) This value can be used to reference the device under `/dev/disk/by-id/`.
- 

The `initialize_params` block supports:

- `description` - (Optional) A description of the boot disk.
- `size` - (Optional) The size of the disk in GB.
- `type` - (Optional) The disk type.
- `image_id` - (Optional) The disk image to initialize this disk from.
- `snapshot_id` - (Optional) The snapshot to initialize this disk from.

**NOTE:** `image_id` or `snapshot_id` must be specified.

---

The `deploy_policy` block supports:

- `max_unavailable` - (Required) The maximum number of running instances that can be taken offline (stopped or deleted) at the same time during the update process.
  - `max_expansion` - (Required) The maximum number of instances that can be temporarily allocated above the group's target size during the update process.
- 
- `max_deleting` - (Optional) The maximum number of instances that can be deleted at the same time.
  - `max_creating` - (Optional) The maximum number of instances that can be created at the same time.
  - `startup_duration` - (Optional) The amount of time in seconds to allow for an instance to start. Instance will be considered up and running (and start receiving traffic) only after the `startup_duration` has elapsed and all health checks are passed.
-

The `scale_policy` block supports:

- `fixed_scale` - (Optional) The fixed scaling policy of the instance group. The structure is documented below.
- 

The `fixed_scale` block supports:

- `size` - (Required) The number of instances in the instance group.

## » Attributes Reference

In addition to the arguments listed above, the following computed attributes are exported:

- `id` - The ID of the instance group.
- `created_at` - The instance group creation timestamp.

## » `yandex_compute_snapshot`

Creates a new snapshot of a disk. For more information, see the official documentation.

## » Example Usage

```
resource "yandex_compute_snapshot" "default" {
  name          = "test-snapshot"
  source_disk_id = "test_disk_id"

  labels = {
    my-label = "my-label-value"
  }
}
```

## » Argument Reference

The following arguments are supported:

- `source_disk_id` - (Required) ID of the disk to create a snapshot from.
- 
- `name` - (Optional) A name for the resource.

- **description** - (Optional) Description of the resource.
- **folder\_id** - (Optional) The ID of the folder that the resource belongs to. If it is not provided, the default provider folder is used.
- **labels** - (Optional) A set of key/value label pairs to assign to the snapshot.

## » Attributes Reference

In addition to the arguments listed above, the following computed attributes are exported:

- **disk\_size** - Size of the disk when the snapshot was created, specified in GB.
- **storage\_size** - Size of the snapshot, specified in GB.
- **created\_at** - Creation timestamp of the snapshot.

## » Import

A snapshot can be imported using the `id` of the resource, e.g.

```
$ terraform import yandex_compute_snapshot.disk-snapshot snapshot_id
```

## » yandex\_iam\_service\_account

Allows management of a Yandex.Cloud IAM service account. To assign roles and permissions, use the `yandex_iam_service_account_iam_binding`, `yandex_iam_service_account_iam_member` and `yandex_iam_service_account_iam_policy` resources.

## » Example Usage

This snippet creates a service account.

```
resource "yandex_iam_service_account" "sa" {
  name      = "VM Manager"
  description = "service account to manage VMs"
}
```

## » Argument Reference

The following arguments are supported:

- **name** - (Optional) Name of the service account. Can be updated without creating a new resource.
- **description** - (Optional) Description of the service account.
- **folder\_id** - (Optional) ID of the folder that the service account will be created in. Defaults to the provider folder configuration.

## » Import

A service account can be imported using the `id` of the resource, e.g.

```
$ terraform import yandex_iam_service_account.sa account_id
```

## » IAM policy for a service account

When managing IAM roles, you can treat a service account either as a resource or as an identity. This resource is used to add IAM policy bindings to a service account resource to configure permissions that define who can edit the service account.

There are three different resources that help you manage your IAM policy for a service account. Each of these resources is used for a different use case:

- `yandex_iam_service_account_iam_policy`: Authoritative. Sets the IAM policy for the service account and replaces any existing policy already attached.
- `yandex_iam_service_account_iam_binding`: Authoritative for a given role. Updates the IAM policy to grant a role to a list of members. Other roles within the IAM policy for the service account are preserved.
- `yandex_iam_service_account_iam_member`: Non-authoritative. Updates the IAM policy to grant a role to a new member. Other members for the role of the service account are preserved.

**Note:** `yandex_iam_service_account_iam_policy` **cannot** be used in conjunction with `yandex_iam_service_account_iam_binding` and `yandex_iam_service_account_iam_member` or they will conflict over what your policy should be.

**Note:** `yandex_iam_service_account_iam_binding` resources **can be** used in conjunction with `yandex_iam_service_account_iam_member` resources **only if** they do not grant privileges to the same role.

## » `yandex_service_account_iam_binding`

```
resource "yandex_iam_service_account_iam_binding" "admin-account-iam" {
  service_account_id = "your-service-account-id"
  role                = "admin"

  members = [
    "userAccount:foo_user_id",
  ]
}
```

## » Argument Reference

The following arguments are supported:

- `service_account_id` - (Required) The service account ID to apply a binding to.
- `role` - (Required) The role that should be applied. Only one `yandex_iam_service_account_iam_binding` can be used per role.
- `members` - (Required) Identities that will be granted the privilege in `role`. Each entry can have one of the following values:
  - `userAccount:{user_id}`: A unique user ID that represents a specific Yandex account.
  - `serviceAccount:{service_account_id}`: A unique service account ID.

## » Import

Service account IAM binding resources can be imported using the service account ID and role.

```
$ terraform import yandex_iam_service_account_iam_binding.admin-account-iam "service_account_id:role"
```

## » IAM policy for a service account

When managing IAM roles, you can treat a service account either as a resource or as an identity. This resource is used to add IAM policy bindings to a service account resource to configure permissions that define who can edit the service account.

There are three different resources that help you manage your IAM policy for a service account. Each of these resources is used for a different use case:

- `yandex_iam_service_account_iam_policy`: Authoritative. Sets the IAM policy for the service account and replaces any existing policy already attached.
- `yandex_iam_service_account_iam_binding`: Authoritative for a given role. Updates the IAM policy to grant a role to a list of members. Other roles within the IAM policy for the service account are preserved.
- `yandex_iam_service_account_iam_member`: Non-authoritative. Updates the IAM policy to grant a role to a new member. Other members for the role of the service account are preserved.

**Note:** `yandex_iam_service_account_iam_policy` **cannot** be used in conjunction with `yandex_iam_service_account_iam_binding` and `yandex_iam_service_account_iam_member` or they will conflict over what your policy should be.

**Note:** `yandex_iam_service_account_iam_binding` resources **can be** used in conjunction with `yandex_iam_service_account_iam_member` resources **only if** they do not grant privileges to the same role.

## » `yandex_service_account_iam_member`

```
resource "yandex_iam_service_account_iam_member" "admin-account-iam" {
  service_account_id = "your-service-account-id"
  role                = "admin"
  member              = "userAccount:bar_user_id"
}
```

## » Argument Reference

The following arguments are supported:

- `service_account_id` - (Required) The service account ID to apply a policy to.
- `role` - (Required) The role that should be applied. Only one `yandex_iam_service_account_iam_binding` can be used per role.
- `member` - (Required) Identity that will be granted the privilege in `role`. Entry can have one of the following values:
  - `userAccount:{user_id}`: A unique user ID that represents a specific Yandex account.
  - `serviceAccount:{service_account_id}`: A unique service account ID.

## » Import

Service account IAM member resources can be imported using the service account ID, role and member.

```
$ terraform import yandex_iam_service_account_iam_member.admin-account-iam "service_account,
```

## » IAM policy for a service account

When managing IAM roles, you can treat a service account either as a resource or as an identity. This resource is used to add IAM policy bindings to a service account resource to configure permissions that define who can edit the service account.

There are three different resources that help you manage your IAM policy for a service account. Each of these resources is used for a different use case:

- `yandex_iam_service_account_iam_policy`: Authoritative. Sets the IAM policy for the service account and replaces any existing policy already attached.
- `yandex_iam_service_account_iam_binding`: Authoritative for a given role. Updates the IAM policy to grant a role to a list of members. Other roles within the IAM policy for the service account are preserved.
- `yandex_iam_service_account_iam_member`: Non-authoritative. Updates the IAM policy to grant a role to a new member. Other members for the role of the service account are preserved.

**Note:** `yandex_iam_service_account_iam_policy` **cannot** be used in conjunction with `yandex_iam_service_account_iam_binding` and `yandex_iam_service_account_iam_member` or they will conflict over what your policy should be.

**Note:** `yandex_iam_service_account_iam_binding` resources **can be** used in conjunction with `yandex_iam_service_account_iam_member` resources **only if** they do not grant privileges to the same role.

## » `yandex_service_account_iam_policy`

```
data "yandex_iam_policy" "admin" {
  binding {
    role = "admin"

    members = [
      "userAccount:foobar_user_id",
    ]
  }
}
```



```

}

resource "yandex_iam_service_account_iam_policy" "admin-account-iam" {
  service_account_id = "your-service-account-id"
  policy_data        = "${data.yandex_iam_policy.admin.policy_data}"
}

```

## » Argument Reference

The following arguments are supported:

- **service\_account\_id** - (Required) The service account ID to apply a policy to.
- **members** - (Required) Identities that will be granted the privilege in **role**. Each entry can have one of the following values:
  - **userAccount:{user\_id}**: A unique user ID that represents a specific Yandex account.
  - **serviceAccount:{service\_account\_id}**: A unique service account ID.
- **role** - (Required) The role that should be applied. Only one **yandex\_iam\_service\_account\_iam\_binding** can be used per role.
- **policy\_data** - (Required only by **yandex\_iam\_service\_account\_iam\_policy**) The policy data generated by a **yandex\_iam\_policy** data source.

## » Import

Service account IAM policy resources can be imported using the service account ID.

```
$ terraform import yandex_iam_service_account_iam_policy.admin-account-iam service_account_id
```

## » yandex\_iam\_service\_account\_static\_access\_key

Allows management of a Yandex.Cloud IAM service account static access keys. Generated pair of keys are used to access Yandex Object Storage on behalf of service account.

Before use keys do not forget to assign a proper role to a service account.

## » Example Usage

This snippet creates a service account static access key.

```
resource "yandex_iam_service_account_static_access_key" "sa-static-key" {
  service_account_id = "some_sa_id"
  description         = "static access key for object storage"
}
```

## » Argument Reference

The following arguments are supported:

- **service\_account\_id** - (Required) ID of the service account which is used to get a static key.
- 
- **description** - (Optional) The description of the service account static key.

## » Attributes Reference

In addition to the arguments listed above, the following computed attributes are exported:

- **access\_key** - ID of the static access key.
- **secret\_key** - Private part of generated static access key.
- **created\_at** - Creation timestamp of the static access key.

## » `yandex_resourcemanager_cloud_iam_binding`

Allows creation and management of a single binding within IAM policy for an existing Yandex Resource Manager cloud.

## » Example Usage

```
data "yandex_resourcemanager_cloud" "project1" {
  name = "Project 1"
}

resource "yandex_resourcemanager_cloud_iam_binding" "admin" {
  cloud_id = "${data.yandex_resourcemanager_cloud.project1.id}"
}
```

```

    role = "editor"

    members = [
        "userAccount:some_user_id",
    ]
}

```

## » Argument Reference

The following arguments are supported:

- `cloud_id` - (Required) ID of the cloud to attach the policy to.
- `role` - (Required) The role that should be assigned. Only one `yandex_resourcemanager_cloud_iam_binding` can be used per role.
- `members` - (Required) An array of identities that will be granted the privilege in the `role`. Each entry can have one of the following values:
  - `userAccount:{user_id}`: A unique user ID that represents a specific Yandex account.
  - `serviceAccount:{service_account_id}`: A unique service account ID.

## » Import

IAM binding imports use space-delimited identifiers; first the resource in question and then the role. These bindings can be imported using the `cloud_id` and `role`, e.g.

```
$ terraform import yandex_resourcemanager_cloud_iam_binding.viewer "cloud_id viewer"
```

## » `yandex__resourcemanager__cloud__iam__member`

Allows creation and management of a single member for a single binding within the IAM policy for an existing Yandex Resource Manager cloud.

**Note:** Roles controlled by `yandex_resourcemanager_cloud_iam_binding` should not be assigned using `yandex_resourcemanager_cloud_iam_member`.

## » Example Usage

```
data "yandex_resourcemanager_cloud" "department1" {
```

```

    name = "Department 1"
  }

  resource "yandex_resourcemanager_cloud_iam_member" "admin" {
    cloud_id = "${data.yandex_resourcemanager_cloud.department1.id}"
    role     = "editor"
    member   = "userAccount:user_id"
  }

```

## » Argument Reference

The following arguments are supported:

- `cloud_id` - (Required) ID of the cloud to attach a policy to.
- `role` - (Required) The role that should be assigned.
- `member` - (Required) The identity that will be granted the privilege that is specified in the `role` field. This field can have one of the following values:
  - `userAccount:{user_id}`: A unique user ID that represents a specific Yandex account.
  - `serviceAccount:{service_account_id}`: A unique service account ID.

## » Import

IAM member imports use space-delimited identifiers; the resource in question, the role, and the account. This member resource can be imported using the `cloud_id`, `role`, and `account`, e.g.

```
$ terraform import yandex_resourcemanager_cloud_iam_member.my_project "cloud_id viewer foo@bar.yandexcloud.net"
```

## » yandex\_\_resourcemanager\_\_folder\_\_iam\_\_binding

Allows creation and management of a single binding within IAM policy for an existing Yandex Resource Manager folder.

**Note:** This resource *must not* be used in conjunction with `yandex_resourcemanager_folder_iam_policy` or they will conflict over what your policy should be.

## » Example Usage

```
data "yandex_resourcemanager_folder" "project1" {
```

```

    folder_id = "some_folder_id"
  }

resource "yandex_resourcemanager_folder_iam_binding" "admin" {
  folder_id = "${data.yandex_resourcemanager_folder.project1.id}"

  role = "editor"

  members = [
    "userAccount:some_user_id",
  ]
}

```

## » Argument Reference

The following arguments are supported:

- **folder\_id** - (Required) ID of the folder to attach a policy to.
- **role** - (Required) The role that should be assigned. Only one `yandex_resourcemanager_folder_iam_binding` can be used per role.
- **members** - (Required) An array of identities that will be granted the privilege that is specified in the `role` field. Each entry can have one of the following values:
  - **userAccount:{user\_id}**: An email address that represents a specific Yandex account. For example, `ivan@yandex.ru` or `joe@example.com`.
  - **serviceAccount:{service\_account\_id}**: A unique service account ID.

## » Import

IAM binding imports use space-delimited identifiers; first the resource in question and then the role. These bindings can be imported using the `folder_id` and role, e.g.

```
$ terraform import yandex_resourcemanager_folder_iam_binding.viewer "folder_id viewer"
```

## » yandex\_\_resourcemanager\_\_folder\_\_iam\_\_member

Allows creation and management of a single member for a single binding within the IAM policy for an existing Yandex Resource Manager folder.

**Note:** This resource *must not* be used in conjunction with `yandex_resourcemanager_folder_iam_policy` or they will conflict over what your policy should be. Similarly, roles controlled by `yandex_resourcemanager_folder_iam_binding` should not be assigned using `yandex_resourcemanager_folder_iam_member`.

## » Example Usage

```
data "yandex_resourcemanager_folder" "department1" {
  folder_id = "some_folder_id"
}

resource "yandex_resourcemanager_folder_iam_member" "admin" {
  folder_id = "${data.yandex_resourcemanager_folder.department1.name}"

  role    = "editor"
  member = "userAccount:user_id"
}
```

## » Argument Reference

The following arguments are supported:

- `folder_id` - (Required) ID of the folder to attach a policy to.
- `role` - (Required) The role that should be assigned.
- `member` - (Required) The identity that will be granted the privilege that is specified in the `role` field. This field can have one of the following values:
  - `userAccount:{user_id}`: A unique user ID that represents a specific Yandex account.
  - `serviceAccount:{service_account_id}`: A unique service account ID.

## » Import

IAM member imports use space-delimited identifiers; the resource in question, the role, and the account. This member resource can be imported using the `folder_id`, `role`, and `account`, e.g.

```
$ terraform import yandex_resourcemanager_folder_iam_member.my_project "folder_id viewer fo
```

## » `yandex__folder__iam__policy`

Allows creation and management of the IAM policy for an existing Yandex Resource Manager folder.

### » Example Usage

```
data "yandex_resourcemanager_folder" "project1" {
  folder_id = "my_folder_id"
}

data "yandex_iam_policy" "admin" {
  binding {
    role = "editor"

    members = [
      "userAccount:some_user_id",
    ]
  }
}

resource "yandex_resourcemanager_iam_policy" "folder_admin_policy" {
  folder_id   = "${data.yandex_folder.project1.id}"
  policy_data = "${data.yandex_iam_policy.admin.policy_data}"
}
```

### » Argument Reference

The following arguments are supported:

- `folder_id` - (Required) ID of the folder that the policy is attached to.
- `policy_data` - (Required) The `yandex_iam_policy` data source that represents the IAM policy that will be applied to the folder. This policy overrides any existing policy applied to the folder.

## » `yandex__vpc__network`

Manages a network within the Yandex.Cloud. For more information, see the official documentation.

- How-to Guides
  - Cloud Networking

– VPC Addressing

## » Example Usage

```
resource "yandex_vpc_network" "default" {  
  name = "foobar"  
}
```

## » Argument Reference

The following arguments are supported:

- **name** - (Optional) Name of the network. Provided by the client when the network is created.
- **description** - (Optional) An optional description of this resource. Provide this property when you create the resource.
- **folder\_id** - (Optional) ID of the folder that the resource belongs to. If it is not provided, the default provider folder is used.
- **labels** - (Optional) Labels to apply to this network. A list of key/value pairs.

## » Attributes Reference

In addition to the arguments listed above, the following computed attributes are exported:

- **created\_at** - Creation timestamp of the key.

## » Import

A network can be imported using the `id` of the resource, e.g.

```
$ terraform import yandex_vpc_network.default network_id
```

## » yandex\_\_vpc\_\_route\_\_table

Manages a route table within the Yandex.Cloud. For more information, see the official documentation.

- How-to Guides
  - Cloud Networking



## » Example Usage

```
resource "yandex_vpc_network" "lab-net" {
  name = "lab-network"
}

resource "yandex_vpc_route_table" "lab-rt-a" {
  network_id = "${yandex_vpc_network.lab-net.id}"

  static_route {
    destination_prefix = "10.2.0.0/16"
    next_hop_address   = "172.16.10.10"
  }
}
```

## » Argument Reference

The following arguments are supported:

- **network\_id** - (Required) ID of the network this route table belongs to.
- 
- **name** - (Optional) Name of the route table. Provided by the client when the route table is created.
  - **description** - (Optional) An optional description of the route table. Provide this property when you create the resource.
  - **folder\_id** - (Optional) The ID of the folder to which the resource belongs. If omitted, the provider folder is used.
  - **labels** - (Optional) Labels to assign to this route table. A list of key/value pairs.
  - **static\_route** - (Optional) A list of static route records for the route table. The structure is documented below.

The **static\_route** block supports:

- **destination\_prefix** - Route prefix in CIDR notation.
- **next\_hop\_address** - Address of the next hop.

## » Attributes Reference

- **created\_at** - Creation timestamp of the route table.

## » Import

A route table can be imported using the `id` of the resource, e.g.:

```
$ terraform import yandex_vpc_route_table.default route_table_id
```

## » yandex\_\_vpc\_\_subnet

Manages a subnet within the Yandex.Cloud. For more information, see the official documentation.

- How-to Guides
  - Cloud Networking
  - VPC Addressing

## » Example Usage

```
resource "yandex_vpc_network" "lab-net" {
  name = "lab-network"
}

resource "yandex_vpc_subnet" "lab-subnet-a" {
  v4_cidr_blocks = ["10.2.0.0/16"]
  zone           = "ru-central1-a"
  network_id     = "${yandex_vpc_network.lab-net.id}"
}
```

## » Argument Reference

The following arguments are supported:

- **network\_id** - (Required) ID of the network this subnet belongs to. Only networks that are in the distributed mode can have subnets.
  - **v4\_cidr\_blocks** - (Required) A list of blocks of internal IPv4 addresses that are owned by this subnet. Provide this property when you create the subnet. For example, 10.0.0.0/22 or 192.168.0.0/16. Blocks of addresses must be unique and non-overlapping within a network. Minimum subnet size is /28, and maximum subnet size is /16. Only IPv4 is supported.
  - **zone** - (Required) Name of the Yandex.Cloud zone for this subnet.
- 
- **name** - (Optional) Name of the subnet. Provided by the client when the subnet is created.

- **description** - (Optional) An optional description of the subnet. Provide this property when you create the resource.
- **folder\_id** - (Optional) The ID of the folder to which the resource belongs. If omitted, the provider folder is used.
- **labels** - (Optional) Labels to assign to this subnet. A list of key/value pairs.
- **route\_table\_id** - (Optional) The ID of the route table to assign to this subnet. Assigned route table should belong to the same network as this subnet.

## » Attributes Reference

In addition to the arguments listed above, the following computed attributes are exported:

- **v6\_cidr\_blocks** - An optional list of blocks of IPv6 addresses that are owned by this subnet.

## » Timeouts

This resource provides the following configuration options for timeouts:

- **create** - Default is 3 minute.
- **update** - Default is 3 minute.
- **delete** - Default is 3 minute.

## » Import

A subnet can be imported using the `id` of the resource, e.g.:

```
$ terraform import yandex_vpc_subnet.default subnet_id
```

Creates a network load balancer in the specified folder using the data specified in the config. For more information, see the official documentation.

## » yandex\_lb\_network\_load\_balancer

### » Example Usage

```
resource "yandex_lb_network_load_balancer" "foo" {
  name = "my-netwokt-load-balancer"
```

```

listener {
  name = "my-listener"
  port = 8080
  external_address_spec {
    ip_version = "ipv4"
  }
}

attached_target_group {
  target_group_id = "${yandex_lb_target_group.my-target-group.id}"

  healthcheck {
    name = "http"
    http_options {
      port = 8080
      path = "/ping"
    }
  }
}
}

```

## » Argument Reference

The following arguments are supported:

- **name** - (Optional) Name of the network load balancer. Provided by the client when the network load balancer is created.
- **description** - (Optional) An optional description of the network load balancer. Provide this property when you create the resource.
- **folder\_id** - (Optional) The ID of the folder to which the resource belongs. If omitted, the provider folder is used.
- **labels** - (Optional) Labels to assign to this network load balancer. A list of key/value pairs.
- **region\_id** - (Optional) ID of the availability zone where the network load balancer resides. The default is 'ru-central1'.
- **type** - (Optional) Type of the network load balancer. Only external network load balancers are currently available. The default is 'external'.
- **attached\_target\_group** - (Optional) An AttachedTargetGroup resource. The structure is documented below.
- **listener** - (Optional) Listener specification that will be used by a network load balancer. The structure is documented below.

---

The **attached\_target\_group** block supports:

- **target\_group\_id** - (Required) ID of the target group.
  - **healthcheck** - (Required) A HealthCheck resource. The structure is documented below.
- 

The **healthcheck** block supports:

- **name** - (Required) Name of the health check. The name must be unique for each target group that attached to a single load balancer.
- **interval** - (Optional) The interval between health checks. The default is 2 seconds.
- **timeout** - (Optional) Timeout for a target to return a response for the health check. The default is 1 second.
- **unhealthy\_threshold** - (Optional) Number of failed health checks before changing the status to **UNHEALTHY**. The default is 2.
- **healthy\_threshold** - (Optional) Number of successful health checks required in order to set the **HEALTHY** status for the target.
- **http\_options** - (Optional) Options for HTTP health check. The structure is documented below.
- **tcp\_options** - (Optional) Options for TCP health check. The structure is documented below.

**NOTE:** One of **http\_options** or **tcp\_options** should be specified.

---

The **http\_options** block supports:

- **port** - (Required) Port to use for HTTP health checks.
  - **path** - (Optional) URL path to set for health checking requests for every target in the target group. For example **/ping**. The default path is **/**.
- 

The **tcp\_options** block supports:

- **port** - (Required) Port to use for TCP health checks.
- 

The **listener** block supports:

- **name** - (Required) Name of the listener. The name must be unique for each listener on a single load balancer.

- **port** - (Required) Port for incoming traffic.
- **target\_port** - (Optional) Port of a target. The default is the same as listener's port.
- **protocol** - (Optional) Protocol for incoming traffic. Only tcp network load balancers are currently available.
- **external\_address\_spec** - (Optional) External IP address specification. The structure is documented below.

---

The **external\_address\_spec** block supports:

- **address** - (Optional) Public IP address for a listener. IP address will be allocated if it wasn't been set.
- **ip\_version** - (Optional) IP version of the addresses that the load balancer works with. Only ipv4 is currently available.

## » Attributes Reference

In addition to the arguments listed above, the following computed attributes are exported:

- **id** - The ID of the network load balancer.
- **created\_at** - The network load balancer creation timestamp.

## » Timeouts

This resource provides the following configuration options for timeouts:

- **create** - Default is 5 minute.
- **update** - Default is 5 minute.
- **delete** - Default is 5 minute.

## » Import

A network load balancer can be imported using the **id** of the resource, e.g.:

```
$ terraform import yandex_lb_network_load_balancer.default network_load_balancer_id
```

Creates a target group in the specified folder and adds the specified targets to it. For more information, see the official documentation.

## » `yandex_lb_target_group`

### » Example Usage

```
resource "yandex_lb_target_group" "foo" {
  name      = "my-target-group"
  region_id = "ru-central1"

  target {
    subnet_id = "${yandex_vpc_subnet.my-subnet.id}"
    address   = "${yandex_compute_instance.my-instance-1.network_interface.0.ip_address}"
  }

  target {
    subnet_id = "${yandex_vpc_subnet.my-subnet.id}"
    address   = "${yandex_compute_instance.my-instance-2.network_interface.0.ip_address}"
  }
}
```

### » Argument Reference

The following arguments are supported:

- **name** - (Optional) Name of the target group. Provided by the client when the target group is created.
- **description** - (Optional) An optional description of the target group. Provide this property when you create the resource.
- **folder\_id** - (Optional) The ID of the folder to which the resource belongs. If omitted, the provider folder is used.
- **labels** - (Optional) Labels to assign to this target group. A list of key/value pairs.
- **region\_id** - (Optional) ID of the availability zone where the target group resides. The default is 'ru-central1'.
- **target** - (Optional) A Target resource. The structure is documented below.

---

The **target** block supports:

- **address** - (Required) IP address of the target.

- **subnet\_id** - (Required) ID of the subnet that targets are connected to. All targets in the target group must be connected to the same subnet within a single availability zone.

## » Attributes Reference

In addition to the arguments listed above, the following computed attributes are exported:

- **id** - The ID of the target group.
- **created\_at** - The target group creation timestamp.

## » Timeouts

This resource provides the following configuration options for timeouts:

- **create** - Default is 5 minute.
- **update** - Default is 5 minute.
- **delete** - Default is 5 minute.

## » Import

A target group can be imported using the **id** of the resource, e.g.:

```
$ terraform import yandex_lb_target_group.default target_group_id
```