» exoscale_compute_template

Provides information on an Compute template for use in other resources such as a `exoscale_compute` resource.

» Example Usage

```
locals {
  zone = "ch-gva-2"
}

data "exoscale_compute_template" "ubuntu" {
  zone = "${local.zone}"
  name = "Linux Ubuntu 18.04 LTS 64-bit"
}

resource "exoscale_compute" "my_server" {
  zone         = "${local.zone}"
  display_name = "my server"
  template_id  = "${data.exoscale_compute_template.ubuntu.id}"
  disk_size    = 10
  key_pair     = "my key"
}
```

» Argument Reference

- `zone` - (Required) The name of the zone where to look for the Compute template.
- `name` - The name of the Compute template.
- `id` - The ID of the Compute template.
- `filter` - A Compute template search filter, must be either `featured` (official Exoscale templates), `community` (community-contributed templates) or `mine` (custom templates private to my organization). Default is `featured`.

» Attributes Reference

The following attributes are exported:

- `id` - ID of the template
- `name` - Name of the template
- `username` - Username to use to log into a Compute Instance based on this template

» exoscale_affinity

Provides an Exoscale Anti-Affinity Group. This can be used to create and delete Anti-Affinity Groups.

» Example Usage

```
resource "exoscale_affinity" "cluster" {
  name        = "cluster"
  description = "HA Cluster"
  type        = "host anti-affinity"
}
```

» Argument Reference

- `name` - (Required) The name of the Anti-Affinity Group.
- `description` - A free-form text describing the Anti-Affinity Group purpose.
- `type` - The type of the Anti-Affinity Group (`host anti-affinity` is the only supported value).

» Attributes Reference

The following attributes are exported:

- `id` - The ID of the Anti-Affinity Group.
- `virtual_machine_ids` - The IDs of the Compute instance resources member of the Anti-Affinity Group.

» Import

An existing Anti-Affinity Group can be imported as a resource by name or ID:

```
# By name
$ terraform import exoscale_affinity.mygroup mygroup

# By ID
$ terraform import exoscale_affinity.mygroup eb556678-ec59-4be6-8c54-0406ae0f6da6
```

» exoscale_compute

Provides an Exoscale Compute instance resource. This can be used to create, modify, and delete Compute instances.

» Example Usage

```
data "exoscale_compute_template" "ubuntu" {
  zone = "ch-gva-2"
  name = "Linux Ubuntu 18.04 LTS 64-bit"
}

resource "exoscale_compute" "mymachine" {
  zone         = "ch-gva-2"
  display_name = "mymachine"
  template_id  = "${data.exoscale_compute_template.ubuntu.id}"
  size         = "Medium"
  disk_size    = 10
  key_pair     = "me@mymachine"
  state        = "Running"

  affinity_groups = []
  security_groups = ["default"]

  ip6 = false

  user_data = <<EOF
#cloud-config
manage_etc_hosts: localhost
EOF

  tags = {
    production = "true"
  }

  timeouts {
    create = "60m"
    delete = "2h"
  }
}
```

» Argument Reference

- `zone` - (Required) The name of the zone to deploy the Compute instance into.
- `display_name` - (Required) The displayed name of the Compute instance. Note: This value is also used to set the OS' *hostname* during creation, so the value can only contain alphanumeric and hyphen ("-") characters; it can be changed to any character during a later update.
- `template` - (Required) The name of the Compute instance template. Only *featured* templates are available, if you want to reference *custom templates* use the `template_id` attribute instead.
- `template_id` - (Required) The ID of the Compute instance template. Usage of the `compute_template` data source is recommended.
- `size` - (Required) The Compute instance size, e.g. `Tiny`, `Small`, `Medium`, `Large` etc.
- `disk_size` - (Required) The Compute instance root disk size in GiB (at least `10`).
- `key_pair` - (Required) The name of the SSH key pair to be installed.
- `user_data` - A cloud-init configuration. Whenever possible don't base64-encode neither gzip it yourself, as this will be automatically taken care of on your behalf by the provider.
- `keyboard` - The keyboard layout configuration (at creation time only). Supported values are: `de`, `de-ch`, `es`, `fi`, `fr`, `fr-be`, `fr-ch`, `is`, `it`, `jp`, `nl-be`, `no`, `pt`, `uk`, `us`.
- `state` - The state of the Compute instance, e.g. `Running` or `Stopped`
- `affinity_groups` - A list of Anti-Affinity Group names (at creation time only; conflicts with `affinity_group_ids`).
- `affinity_group_ids` - A list of Anti-Affinity Group IDs (at creation time only; conflicts with `affinity_groups`).
- `security_groups` - A list of Security Group names (conflicts with `security_group_ids`).
- `security_group_ids` - A list of Security Group IDs (conflicts with `security_groups`).
- `ip4` - Boolean controlling if IPv4 is enabled (only supported value is `true`).
- `ip6` - Boolean controlling if IPv6 is enabled.
- `tags` - A dictionary of tags (key/value).

» Attributes Reference

The following attributes are exported:

- `name` - The name of the Compute instance (*hostname*).
- `username` - The user to use to connect to the Compute instance with SSH. If you've referenced a *custom template* in the resource, use the `compute_template` data source `username` attribute instead.

- `password` - The initial Compute instance password and/or encrypted password.
- `ip_address` - The IP address of the Compute instance main network interface.
- `ip6_address` - The IPv6 address of the Compute instance main network interface.

» `remote-exec` provisioner usage

If you wish to log to a `exoscale_compute` resource using the `remote-exec` provisioner, make sure to explicity set the SSH `user` setting to connect to the instance to the actual template username returned by the `exoscale_compute_template` data source:

```
data "exoscale_compute_template" "ubuntu" {
  zone = "ch-gva-2"
  name = "Linux Ubuntu 18.04 LTS 64-bit"
}

resource "exoscale_compute" "mymachine" {
  zone         = "ch-gva-2"
  display_name = "mymachine"
  template_id  = "${data.exoscale_compute_template.ubuntu.id}"
  size         = "Medium"
  disk_size    = 10
  key_pair     = "me@mymachine"
  state        = "Running"

  provisioner "remote-exec" {
    connection {
      type = "ssh"
      host = "${self.ip_address}"
      user = "${data.exoscale_compute_template.ubuntu.username}"
    }
  }
}
```

» Import

An existing Compute instance can be imported as a resource by name or ID. Importing a Compute instance imports the `exoscale_compute` resource as well as related `exoscale_secondary_ipaddress` and `exoscale_nic` resources.

```
# By name
$ terraform import exoscale_compute.vm1 vm1
```

```
# By ID
$ terraform import exoscale_compute.vm1 eb556678-ec59-4be6-8c54-0406ae0f6da6
```

» exoscale_domain

Provides an Exoscale DNS Domain resource. This can be used to create and delete DNS Domains.

» Usage example

```
resource "exoscale_domain" "example" {
  name = "example.net"
}
```

» Argument Reference

- `name` - (Required) The name of the DNS Domain.

» Attributes Reference

The following attributes are exported:

- `token` - A security token that can be used as an alternative way to manage DNS Domains via the Exoscale API.
- `state` - The state of the DNS Domain.
- `auto_renew` - Boolean indicating that the DNS Domain has automatic renewal enabled.
- `expires_on` - The date of expiration of the DNS Domain, if known.

» Import

An existing DNS Domain can be imported as a resource by name:

```
$ terraform import exoscale_domain.example example.net
```

NOTE: importing a `exoscale_domain` resource will also import all related [`exoscale_domain_records`][domainrec] resources (except `NS` and `SOA`).

» exoscale_domain_record

Provides an Exoscale DNS Domain Record resource. This can be used to create, modify, and delete DNS Domain Records.

» Usage example

```
resource "exoscale_domain" "example" {
  name = "example.net"
}

resource "exoscale_domain_record" "myserver" {
  domain      = "${exoscale_domain.example.id}"
  name        = "myserver"
  record_type = "A"
  content     = "1.2.3.4"
}

resource "exoscale_domain_record" "myserver_alias" {
  domain      = "${exoscale_domain.example.id}"
  name        = "myserver-new"
  record_type = "CNAME"
  content     = "${exoscale_domain_record.myserver.hostname}"
}
```

» Argument Reference

- `domain` - (Required) The name of the `exoscale_domain` to create the record into.
- `name` - (Required) The name of the DNS Domain Record.
- `record_type` - (Required) The type of the DNS Domain Record. Supported values are: `A`, `AAAA`, `ALIAS`, `CAA`, `CNAME`, `HINFO`, `MX`, `NAPTR`, `NS`, `POOL`, `SPF`, `SRV`, `SSHFP`, `TXT`, `URL`.
- `content` - (Required) The value of the DNS Domain Record.
- `ttl` - The Time To Live of the DNS Domain Record.
- `prio` - The priority of the DNS Domain Record (for types that support it).

» Attributes Reference

The following attributes are exported:

- `hostname` - The DNS Domain Record's *Fully Qualified Domain Name* (FQDN), useful for linking `A` records into `CNAME`.

» Import

An existing DNS Domain Record can be imported as a resource by ID:

```
$ terraform import exoscale_domain_record.www 12480484
```

NOTE: importing an existing `exoscale_domain` resource also imports linked `exoscale_domain_record` resources.

» exoscale_instance_pool

Provides an Exoscale Instance Pool resource. This can be used to create, modify, and delete Instance Pools.

» Example Usage

```
resource "exoscale_ssh_keypair" "key" {
  name = "terraform-mywebapp-keypair"
}

variable "zone" {
  default = "de-fra-1"
}

resource "exoscale_security_group" "web" {
  name = "web"
  description = "Security Group for webapp production"
}

resource "exoscale_network" "web_privnet" {
  zone = "${var.zone}"
  name = "web-privnet"
}

data "exoscale_compute_template" "mywebapp" {
  zone = "${var.zone}"
  name = "mywebapp"
  filter = "mine"
}
```

```
resource "exoscale_instance_pool" "webapp" {
  zone = "${var.zone}"
  name = "webapp"
  template_id = "${data.exoscale_compute_template.mywebbapp.id}"
  size = 3
  service_offering = "Medium"
  disk_size = 50
  description = "This is the production environment for my webapp"
  user_data = "#cloud-config\npackage_upgrade: true\n"
  key_pair = "${exoscale_ssh_keypair.key.name}"

  security_group_ids = [${exoscale_security_group.web.id}]
  network_ids = [${exoscale_network.web_privnet.id}]

  timeouts {
    delete = "10m"
  }
}
```

» Argument Reference

- `zone` - (Required) The name of the zone to deploy the Instance Pool into.
- `name` - (Required) The name of the Instance Pool.
- `template_id` - (Required) (Required) The ID of the instance template to use when creating Compute instances. Usage of the `compute_template` data source is recommended.
- `size` - (Required) The number of Compute instance members the Instance Pool manages.
- `service_offering` - (Required) The managed Compute instances size, e.g. `Tiny`, `Small`, `Medium`, `Large` etc.
- `disk_size` - The managed Compute instances disk size.
- `description` - The description of the Instance Pool.
- `user_data` - A cloud-init configuration to apply when creating Compute instances. Whenever possible don't base64-encode neither gzip it yourself, as this will be automatically taken care of on your behalf by the provider.
- `key_pair` - The name of the SSH key pair to install when creating Compute instances.
- `security_group_ids` - A list of [Security Group][sg] IDs.
- `network_ids` - A list of Private Network IDs.

» Import

An existing Instance Pool can be imported as a resource by name or ID. Importing an Instance Pool imports the `exoscale_instance_pool` resource.

```
# By name
$ terraform import exoscale_instance_pool.pool mypool

# By ID
$ terraform import exoscale_instance_pool.pool eb556678-ec59-4be6-8c54-0406ae0f6da6
```

» exoscale_ipaddress

Provides an Exoscale Elastic IP resource. This can be used to create, update and delete Elastic IPs.

See `exoscale_secondary_ipaddress` for usage with Compute instances.

» Usage example

```
resource "exoscale_ipaddress" "myip" {
  zone = "ch-dk-2"
  tags = {
    usage = "load-balancer"
  }
}
```

Managed EIP:

```
resource "exoscale_ipaddress" "myip" {
  zone                     = "ch-dk-2"
  description              = "My elastic IP for load balancer"
  healthcheck_mode         = "http"
  healthcheck_port         = 8000
  healthcheck_path         = "/status"
  healthcheck_interval     = 5
  healthcheck_timeout      = 2
  healthcheck_strikes_ok   = 2
  healthcheck_strikes_fail = 3
}
```

» Argument Reference

- `zone` - (Required) The name of the zone to create the Elastic IP into.
- `description` - The description of the Elastic IP.
- `healthcheck_mode` - The healthcheck probing mode (must be either `tcp` or `http`).
- `healthcheck_port` - The healthcheck service port to probe (must be between `1` and `65535`).

- `healthcheck_path` - The healthcheck probe HTTP request path (must be specified in `http` mode).
- `healthcheck_interval` - The healthcheck probing interval in seconds (must be between `5` and `300`).
- `healthcheck_timeout` - The time in seconds before considering a healthcheck probing failed (must be between `2` and `60`).
- `healthcheck_strikes_ok` - The number of successful healthcheck probes before considering the target healthy (must be between `1` and `20`).
- `healthcheck_strikes_fail` - The number of unsuccessful healthcheck probes before considering the target unhealthy (must be between `1` and `20`).
- `tags` - A dictionary of tags (key/value).

» Attributes Reference

The following attributes are exported:

- `ip_address` - The Elastic IP address.

» Import

An existing Elastic IP can be imported as a resource by address or ID:

```
# By address
$ terraform import exoscale_ipaddress.myip 159.100.251.224

# By ID
$ terraform import exoscale_ipaddress.myip eb556678-ec59-4be6-8c54-0406ae0f6da6
```

» exoscale_network

Provides an Exoscale Private Network resource. This can be used to create, update and delete Private Networks.

See `exoscale_nic` for usage with Compute instances.

» Usage

```
resource "exoscale_network" "unmanaged" {
  zone             = "ch-gva-2"
  name             = "oob"
  display_text     = "Out-of-band network"
```

```
  tags = {
    ...
  }
}
```

Managed Private Network (~> **NOTE:** this feature is currently only available in the `ch-gva-2` zone):

```
resource "exoscale_network" "managed" {
  zone             = "ch-gva-2"
  name             = "oob"
  display_text     = "Out-of-band network with DHCP"

  start_ip = "10.0.0.20"
  end_ip   = "10.0.0.253"
  netmask  = "255.255.255.0"
}
```

» Argument Reference

- `zone` - (Required) The name of the zone to create the Private Network into.
- `name` - (Required) The name of the Private Network.
- `display_text` - A free-form text describing the Private Network purpose.
- `start_ip` - The first address of IP range used by the DHCP service to automatically assign. Required for *managed* Private Networks.
- `end_ip` - The last address of the IP range used by the DHCP service. Required for *managed* Private Networks.
- `netmask` - The netmask defining the IP network allowed for the static lease (see `exoscale_nic` resource). Required for *managed* Private Networks.
- `tags` - A dictionary of tags (key/value).

» Import

An existing Private Network can be imported as a resource by name or ID:

```
# By name
$ terraform import exoscale_network.net myprivnet

# By ID
$ terraform import exoscale_network.net 04fb76a2-6d22-49be-8da7-f2a5a0b902e1
```

» exoscale_nic

Provides an Exoscale Compute instance Private Network Interface (NIC) resource. This can be used to create, update and delete Compute instance NICs.

» Usage

```
resource "exoscale_compute" "vm1" {
  ...
}

resource "exoscale_network" "oob" {
  ...
}

resource "exoscale_nic" "oob" {
  compute_id = "${exoscale_compute.vm1.id}"
  network_id = "${exoscale_network.oob.id}"
}
```

» Argument Reference

- `compute_id` - (Required) The Compute instance ID.
- `network_id` - (Required) The Private Network ID.
- `ip_address` - The IP address to request as static DHCP lease if the NIC is attached to a *managed* Private Network (see the `exoscale_network` resource).

» Attributes Reference

The following attributes are exported:

- `mac_address` - The physical address (MAC) of the Compute instance NIC.

» Import

This resource is automatically imported when importing an `exoscale_compute` resource.

» exoscale_security_group

Provides an Exoscale Security Group resource. This can be used to create and delete Security Groups.

» Example usage

```
resource "exoscale_security_group" "web" {
  name        = "web"
  description = "Webservers"

  tags = {
    kind = "web"
  }
}
```

» Argument Reference

The following attributes are exported:

- `name` - (Required) The name of the Security Group.
- `description` - A free-form text describing the Anti-Affinity Group purpose.
- `tags` - A dictionary of tags (key/value).

» Import

An existing Security Group can be imported as a resource by name or ID:

```
# By name
$ terraform import exoscale_security_group.http http

# By ID
$ terraform import exoscale_security_group.http eb556678-ec59-4be6-8c54-0406ae0f6da6
```

NOTE: Importing a `exoscale_security_group` resource also imports related `exoscale_security_group_rule` resources.

» exoscale_security_group_rule

Provides an Exoscale Security Group Rule resource. This can be used to create and delete Security Group Rules.

» Example usage

```
resource "exoscale_security_group" "webservers" {
  ...
}

resource "exoscale_security_group_rule" "http" {
  security_group_id = "${exoscale_security_group.webservers.id}"
  type              = "INGRESS"
  protocol          = "TCP"
  cidr              = "0.0.0.0/0" # "::/0" for IPv6
  start_port        = 80
  end_port          = 80
}
```

» Argument Reference

- `security_group` - (Required) The Security Group name the rule applies to.
- `security_group_id` - (Required) The Security Group ID the rule applies to.
- `type` - (Required) The traffic direction to match (`INGRESS` or `EGRESS`).
- `protocol` - (Required) The network protocol to match. Supported values are: `TCP`, `UDP`, `ICMP`, `ICMPv6`, `AH`, `ESP`, `GRE`, `IPIP` and `ALL`.
- `description` - A free-form text describing the Security Group Rule purpose.
- `start_port`/`end_port` - A `TCP`/`UDP` port range to match.
- `icmp_type`/`icmp_code` - An `ICMP`/`ICMPv6` type/code to match.
- `cidr` - A source (for ingress)/destination (for egress) IP subnet to match (conflicts with `user_security_group`).
- `user_security_group_id` - A source (for ingress)/destination (for egress) Security Group ID to match (conflicts with `cidr`).
- `user_security_group` - A source (for ingress)/destination (for egress) Security Group name to match (conflicts with `cidr`).

» Attributes Reference

The following attributes are exported:

- `security_group` - The name of the Security Group the rule applies to.
- `security_group_id` - The ID of the Security Group the rule applies to.
- `user_security_group` - The name of the source (for ingress)/destination (for egress) Security Group to match.

» Import

This resource is automatically imported when importing an `exoscale_security_group` resource.

» exoscale_security_group_rules

Provides a resource for assigning multiple rules to an existing Exoscale Security Group.

» Example usage

```
resource "exoscale_security_group" "webservers" {
  ...
}

resource "exoscale_security_group_rules" "admin" {
  security_group = "${exoscale_security_group.webservers.name}"

  ingress {
    protocol                = "ICMP"
    icmp_type               = 8
    user_security_group_list = ["bastion"]
  }

  ingress {
    protocol                = "TCP"
    ports                   = ["22"]
    user_security_group_list = ["bastion"]
  }
}

resource "exoscale_security_group_rules" "web" {
  security_group_id = "${exoscale_security_group.webservers.id}"

  ingress {
    protocol  = "TCP"
    ports     = ["80", "443"]
    cidr_list = ["0.0.0.0/0", "::/0"]
  }
}
```

» Argument Reference

The following attributes are exported:

- `security_group` - (Required) The Security Group name the rules apply to.
- `security_group_id` - (Required) The Security Group ID the rules apply to.

`egress` and `ingress` support the following:

- `protocol` - (Required) The network protocol to match. Supported values are: `TCP`, `UDP`, `ICMP`, `ICMPv6`, `AH`, `ESP`, `GRE`, `IPIP` and `ALL`.
- `description` - A free-form text describing the Security Group Rule purpose.
- `ports` - A list of ports or port ranges (`start_port-end_port`).
- `icmp_type/icmp_code` - An `ICMP/ICMPv6` type/code to match.
- `cidr_list` - A list of source (for ingress)/destination (for egress) IP subnet to match (conflicts with `user_security_group`).
- `user_security_group_list` - A source (for ingress)/destination (for egress) of the traffic identified by a security group

» Attributes Reference

The following attributes are exported:

- `security_group` - The name of the Security Group the rules apply to.
- `security_group_id` - The ID of the Security Group the rules apply to.

» exoscale_secondary_ipaddress

Provides a resource for assigning an existing Exoscale Elastic IP to a Compute instance.

NOTE: The network interfaces of the Compute instance itself still have to be configured accordingly (unless using a *managed* Elastic IP).

» Secondary IP Address

```
resource "exoscale_compute" "vm1" {
  ...
}

resource "exoscale_ipaddress" "vip" {
  ...
```

```
}

resource "exoscale_secondary_ipaddress" "vip" {
  compute_id = "${exoscale_compute.vm1.id}"
  ip_address = "${exoscale_ipaddress.vip.ip_address}"
}
```

» Argument Reference

- `compute_id` - (Required) The ID of the Compute instance.
- `ip_address` - (Required) The Elastic IP address to assign.

» Attributes Reference

The following attributes are exported:

- `nic_id` - The ID of the NIC.
- `network_id` - The ID of the Network the Compute instance NIC is attached to.

» Import

This resource is automatically imported when importing an `exoscale_compute` resource.

» exoscale_ssh_keypair

Provides an Exoscale SSH Keypair resource. This can be used to create and delete SSH Keypairs.

» Example Usage

```
resource "exoscale_ssh_keypair" "admin" {
  name       = "admin"
  public_key = "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDGRY..."
}
```

» Argument Reference

- `name` - (Required) The name of the SSH Keypair.
- `public_key` - A SSH public key that will be copied into the instances at **first** boot. If not provided, a SSH keypair is generated and the is saved locally (see the `private_key` attribute).

» Attributes Reference

The following attributes are exported:

- `fingerprint` - The unique identifier of the SSH Keypair.
- `public_key` - The SSH public key generated if none was provided.
- `private_key` - The SSH private key generated if no public key was provided.

» Import

An existing SSH Keypair can be imported as a resource by name:

```
$ terraform import exoscale_ssh_keypair.mykey my-key
```