cloudbase-init Documentation

Release 1.1.2

Cloudbase Solutions Srl

Jun 22, 2020

Contents

1 Intro **3**
- 1.1 Portable cloud initialization service . . . 3
- 1.2 Binaries . . . 3

2 Tutorial **5**
- 2.1 Sysprepping . . . 5
- 2.2 Configuration file . . . 5
- 2.3 File execution . . . 6

3 Services **9**
- 3.1 Configuring available services . . . 9
- 3.2 OpenStack (web API) . . . 9
- 3.3 OpenStack (configuration drive) . . . 10
- 3.4 NoCloud configuration drive . . . 11
- 3.5 Amazon EC2 . . . 12
- 3.6 Apache CloudStack . . . 12
- 3.7 OpenNebula Service . . . 13
- 3.8 Ubuntu MaaS . . . 13
- 3.9 Open Virtualization Format (OVF) . . . 14
- 3.10 Packet Service . . . 14
- 3.11 Azure Service . . . 15
- 3.12 Empty Metadata Service . . . 16
- 3.13 VMware GuestInfo Service . . . 16
- 3.14 Google Compute Engine Service . . . 17

4 Plugins **19**
- 4.1 Configuring selected plugins . . . 19
- 4.2 Setting hostname (MAIN) . . . 20
- 4.3 Creating user (MAIN) . . . 20
- 4.4 Setting password (MAIN) . . . 21
- 4.5 Static networking (MAIN) . . . 21
- 4.6 Saving public keys (MAIN) . . . 21
- 4.7 Volume expanding (MAIN) . . . 22
- 4.8 WinRM listener (MAIN) . . . 22
- 4.9 WinRM certificate (MAIN) . . . 22
- 4.10 Local Scripts execution (MAIN) . . . 23
- 4.11 Licensing (MAIN) . . . 23

4.12 Clock synchronization (PRE_NETWORKING) . . . 24
4.13 MTU customization (PRE_METADATA_DISCOVERY) . . . 24
4.14 User data (MAIN) . . . 24
4.15 Trim Config (MAIN) . . . 24
4.16 San Policy Config (MAIN) . . . 25
4.17 RDP Settings Config (MAIN) . . . 25
4.18 RDP Post Certificate Thumbprint (MAIN) . . . 25
4.19 Page Files (MAIN) . . . 25
4.20 Display Idle Timeout Config (MAIN) . . . 25
4.21 Boot Status Policy Config (MAIN) . . . 26
4.22 BCD Config (MAIN) . . . 26
4.23 Ephemeral Disk Config (MAIN) . . . 26
4.24 Windows Auto Updates (MAIN) . . . 26
4.25 Server Certificates (MAIN) . . . 27
4.26 Azure Guest Agent (MAIN) . . . 27

5 Userdata **29**
5.1 PEM certificate . . . 29
5.2 Batch . . . 29
5.3 PowerShell . . . 29
5.4 Bash . . . 30
5.5 Python . . . 30
5.6 EC2 format . . . 30
5.7 Cloud config . . . 30
5.8 Multi-part content . . . 33
5.9 Sysnativeness . . . 33

6 Configuration options reference **35**
6.1 DEFAULT . . . 35
6.2 azure . . . 46
6.3 cloudstack . . . 46
6.4 config_drive . . . 47
6.5 ec2 . . . 48
6.6 gce . . . 49
6.7 maas . . . 49
6.8 openstack . . . 51
6.9 ovf . . . 51
6.10 packet . . . 52
6.11 vmwareguestinfo . . . 52

7 Indices and tables **53**

Index **55**

Contents:

CHAPTER 1

Intro

The open source project **cloudbase-init** is a service conceived and maintained by Cloudbase Solutions Srl, currently working on NT systems. It was designed to initialize and configure guest operating systems under OpenStack, OpenNebula, CloudStack, MaaS and many others. Under Cloudbase page, stable and beta installers can be found and the service itself is very easy to configure through configuration files. It can also customize instances based on user input like local scripts and data.

More details on how you can use this can be found under *Tutorial*.

1.1 Portable cloud initialization service

The main goal of this project is to provide guest cloud initialization for *Windows* and other operating systems. The architecture of the project is highly flexible and allows extensions for additional clouds and plugins.

There's no limitation in the type of supported hypervisors. This service can be used on instances running on Hyper-V, KVM, Xen, ESXi etc.

1.2 Binaries

Stable installers:

- https://www.cloudbase.it/downloads/CloudbaseInitSetup_Stable_x64.msi
- https://www.cloudbase.it/downloads/CloudbaseInitSetup_Stable_x86.msi

Beta installers:

- https://www.cloudbase.it/downloads/CloudbaseInitSetup_x64.msi
- https://www.cloudbase.it/downloads/CloudbaseInitSetup_x86.msi

Use a x64 installer on 64 bit versions of Windows and the x86 one exclusively on 32 bit versions.

CHAPTER 2

Tutorial

First, download your desired type of installer from *here*, then install it and fill in configuration options which suits you best. Based on the current selected *cloudbase-init* installer architecture, it'll be available under *C:\Program Files* or *C:\Program Files (x86)* as **Cloudbase Solutions\Cloudbase-Init** directory. There, are located some folders of interest like:

- bin - Executable files and other binaries.
- conf - Configuration files holding miscellaneous options.
- log - Here are the cloudbase-init logs.
- LocalScripts - User supplied *scripts*.
- Python - Bundle of executable and library files to support Python scripts and core execution.

After install, cloudbase-init acts like a 2-step service which will read metadata using *Services* and will pass that to the executing *Plugins*, this way configuring all the supported things. Depending on the platform, some plugins may request reboots.

2.1 Sysprepping

The System Preparation (Sysprep) tool prepares an installation of Windows for duplication, auditing, and customer delivery. Duplication, also called imaging, enables you to capture a customized Windows image that you can reuse throughout an organization. The Sysprep phase uses the "Unattend.xml" which implies the service to run using the "cloudbase-init-unattend.conf" configuration file.

2.2 Configuration file

In the chosen installation path, under the *conf* directory, are present two config files named "cloudbase-init.conf" and "cloudbase-init-unattend.conf". These can hold various config options for picking up the desired available services and plugins ready for execution and also customizing user experience.

Explained example of configuration file:

```
[DEFAULT]
# What user to create and in which group(s) to be put.
username=Admin
groups=Administrators
inject_user_password=true  # Use password from the metadata (not random).
# Which devices to inspect for a possible configuration drive (metadata).
config_drive_raw_hhd=true
config_drive_cdrom=true
# Path to tar implementation from Ubuntu.
bsdtar_path=C:\Program Files (x86)\Cloudbase Solutions\Cloudbase-Init\bin\bsdtar.exe
# Logging debugging level.
verbose=true
debug=true
# Where to store logs.
logdir=C:\Program Files (x86)\Cloudbase Solutions\Cloudbase-Init\log\
logfile=cloudbase-init-unattend.log
default_log_levels=comtypes=INFO,suds=INFO,iso8601=WARN
logging_serial_port_settings=
# Enable MTU and NTP plugins.
mtu_use_dhcp_config=true
ntp_use_dhcp_config=true
# Where are located the user supplied scripts for execution.
local_scripts_path=C:\Program Files (x86)\Cloudbase Solutions\Cloudbase-
↪Init\LocalScripts\
# Services that will be tested for loading until one of them succeeds.
metadata_services=cloudbaseinit.metadata.services.configdrive.ConfigDriveService,
                  cloudbaseinit.metadata.services.httpservice.HttpService,
                  cloudbaseinit.metadata.services.ec2service.EC2Service,
                  cloudbaseinit.metadata.services.maasservice.MaaSHttpService
# What plugins to execute.
plugins=cloudbaseinit.plugins.common.mtu.MTUPlugin,
        cloudbaseinit.plugins.common.sethostname.SetHostNamePlugin
# Miscellaneous.
allow_reboot=false    # allow the service to reboot the system
stop_service_on_exit=false
```

The "cloudbase-init-unattend.conf" configuration file is similar to the default one and is used by the Sysprepping phase. It was designed for the scenario where the minimum user intervention is required and it only runs the MTU and host name plugins, leaving the image ready for further initialization cases.

More of these explained options are available under the *Services*, *Plugins* and *Userdata* documentation.

A complete list of config options can be found at *Configuration options reference*.

2.3 File execution

Cloudbase-init has the ability to execute user provided scripts, usually found in the default path *C:\Program Files (x86)\Cloudbase Solutions\Cloudbase-Init\LocalScripts*, through a specific *plugin* for doing it. Depending on the platform used, the files should be valid PowerShell, Python, Batch or Bash scripts. The userdata can be also a PEM certificate, in a cloud-config format or a MIME content. The user data plugin is capable of executing various script types and exit code value handling.

Based on their exit codes, you can instruct the system to reboot or even re-execute the plugin on the next boot:

- 1001 - reboot and don't run the plugin again on next boot

- 1002 - don't reboot now and run the plugin again on next boot
- 1003 - reboot and run the plugin again on next boot

CHAPTER 3

Services

A **metadata service** has the role of getting the guest provided data (configuration information) and exposing it to the *Plugins* for a general and basic initialization of the instance. These sub-services can change their behavior according to custom configuration options documented below.

3.1 Configuring available services

Any of these classes can be specified manually in the configuration file under *metadata_services* option. Based on this option, the service loader will search across these providers in the defined order and load the first one that is available.

For more details on doing this, see *configuration* file in *Tutorial*.

3.2 OpenStack (web API)

class cloudbaseinit.metadata.services.httpservice.**HttpService**

A complete service which supports password related capabilities and can be usually accessed at the http://169.254.169.254/ magic URL. The magic URL can be customized using the *metadata_base_url* config option. A default value of *True* for *add_metadata_private_ip_route* option is used to add a route for the IP address to the gateway. This is needed for supplying a bridge between different VLANs in order to get access to the web server.

Metadata version used: *latest*.

Capabilities:

- instance id
- hostname
- public keys

- WinRM authentication certificates
- static network configuration
- admin user password
- post admin user password (only once)
- user data

Config options for *openstack* section:

- metadata_base_url (string: "http://169.254.169.254/")
- add_metadata_private_ip_route (bool: True)
- https_allow_insecure (bool: False)
- https_ca_bundle (string: None)

Config options for *default* section:

- retry_count (integer: 5)
- retry_count_interval (integer: 4)

3.3 OpenStack (configuration drive)

class cloudbaseinit.metadata.services.configdrive.**ConfigDriveService**

This is similar to the web API, but it "serves" its files locally without requiring network access. The data is generally retrieved from a cdrom, vfat or *raw* disks/partitions by enabling selective lookup across different devices. Use the *types* option to specify which types of config drive content the service will search for and also on which devices using the *locations* option.

It will search for metadata:

a. in mounted optical units

b. directly in the physical disk bytes

c. by exploring the physical disk as a vfat drive; which requires *mtools* (specified by the *mtools_path* option in the *Default* section)

This service is usually faster than the HTTP twin, as there is no timeout waiting for the network to be up.

Metadata version used: *latest*.

Capabilities:

- instance id
- hostname
- public keys
- authentication certificates
- static network configuration
- admin user password
- user data

Config options for *config_drive* section:

- raw_hdd (bool: True)

- cdrom (bool: True)
- vfat (bool: True)
- types (list: ["vfat", "iso"])
- locations (list: ["cdrom", "hdd", "partition"])

3.4 NoCloud configuration drive

class cloudbaseinit.metadata.services.nocloudservice.**NoCloudConfigDriveService**

NoCloudConfigDriveService is similar to OpenStack config drive metadata in terms of the medium on which the data is provided (as an attached ISO, partition or disk) and similar to the EC2 metadata in terms of how the metadata files are named and structured.

The metadata is provided on a config-drive (vfat or iso9660) with the label cidata or CIDATA.

The folder structure for NoCloud is:

- /user-data
- /meta-data

The user-data and meta-data files respect the EC2 metadata service format.

Capabilities:

- instance id
- hostname
- public keys
- static network configuration (Debian format)
- user data

Config options for *config_drive* section:

- raw_hdd (bool: True)
- cdrom (bool: True)
- vfat (bool: True)
- types (list: ["vfat", "iso"])
- locations (list: ["cdrom", "hdd", "partition"])

Example metadata:

```
instance-id: windows1
network-interfaces: |
  iface Ethernet0 inet static
  address 10.0.0.2
  network 10.0.0.0
  netmask 255.255.255.0
  broadcast 10.0.0.255
  gateway 10.0.0.1
  hwaddress ether 00:11:22:33:44:55
hostname: windowshost1
```

More information on the NoCloud metadata service specifications can be found here.

3.5 Amazon EC2

class cloudbaseinit.metadata.services.ec2service.**EC2Service**

This is similar to the OpenStack HTTP service but is using a different format for metadata endpoints and has general capabilities.

Metadata version used: *2009-04-04*.

Capabilities:

- instance id
- hostname
- public keys
- user data

Config options for *ec2* section:

- metadata_base_url (string: "http://169.254.169.254/")
- add_metadata_private_ip_route (bool: True)
- https_allow_insecure (bool: False)
- https_ca_bundle (string: None)

Config options for *default* section:

- retry_count (integer: 5)
- retry_count_interval (integer: 4)

Note: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-metadata.html

3.6 Apache CloudStack

class cloudbaseinit.metadata.services.cloudstack.**CloudStack**

Another web-based service which usually uses "10.1.1.1" or DHCP addresses for retrieving content. If no metadata can be found at the *metadata_base_url*, the service will look for the metadata at the DHCP server URL.

Capabilities:

- instance id
- hostname
- public keys
- admin user password
- poll for, post, delete admin user password (each reboot)
- user data

Config options for *cloudstack* section:

- metadata_base_url (string: "http://10.1.1.1/")
- password_server_port (int: 8080)

- add_metadata_private_ip_route (bool: True)
- https_allow_insecure (bool: False)
- https_ca_bundle (string: None)

Config options for *default* section:

- retry_count (integer: 5)
- retry_count_interval (integer: 4)

Note: By design, this service can update the password anytime, so it will cause the *setuserpassword* plugin to run at every boot and by security concerns, the password is deleted right after retrieval and no updating will occur until a new password is available on the server.

3.7 OpenNebula Service

class cloudbaseinit.metadata.services.opennebulaservice.**OpenNebulaService**

The *OpenNebula* provider is related to configuration drive and searches for a specific context file which holds all the available info. The provided details are exposed as bash variables gathered in a shell script.

Capabilities:

- hardcoded instance id to *iid-dsopennebula*
- hostname
- public keys
- static network configuration
- user data

Config options for *default* section:

- retry_count (integer: 5)
- retry_count_interval (integer: 4)

3.8 Ubuntu MaaS

class cloudbaseinit.metadata.services.maasservice.**MaaSHttpService**

This metadata service usually works with instances on baremetal and uses web requests for retrieving the available exposed metadata. It uses OAuth to secure the requests.

Metadata version used: *2012-03-01*.

Capabilities:

- instance id
- hostname
- public keys
- WinRM authentication certificates
- static network configuration

- user data

Config options for *maas* section:

- metadata_base_url (string: None)
- oauth_consumer_key (string: None)
- oauth_consumer_secret (string: None)
- oauth_token_key (string: None)
- oauth_token_secret (string: None)
- https_allow_insecure (bool: False)
- https_ca_bundle (string: None)

Config options for *default* section:

- retry_count (integer: 5)
- retry_count_interval (integer: 4)

Note: By design, the configuration options are set by an agent called curtin which runs the hooks that set the config values. On Windows, these hooks need to be present in the root directory: Windows curtin hooks.

3.9 Open Virtualization Format (OVF)

class cloudbaseinit.metadata.services.ovfservice.**OvfService**

The *OVF* provider searches data from OVF environment ISO transport.

Capabilities:

- instance id (hardcoded to *iid-ovf* if not present)
- hostname
- public keys
- admin user name
- admin user password
- user data

Config options:

- config_file_name (string: "ovf-env.xml")
- drive_label (string: "OVF ENV")
- ns (string: "oe")

3.10 Packet Service

class cloudbaseinit.metadata.services.packet.**PacketService**

Packet metadata service provides the metadata for baremetal servers at the magic URL *https://metadata.packet.net/*.

Capabilities:

- instance id
- hostname
- public keys
- post admin user password (only once)
- user data
- call home on successful provision

Config options for *packet* section:

- metadata_base_url (string: "https://metadata.packet.net/")
- https_allow_insecure (bool: False)
- https_ca_bundle (string: None)

Config options for *default* section:

- retry_count (integer: 5)
- retry_count_interval (integer: 4)

3.11 Azure Service

class cloudbaseinit.metadata.services.azureservice.**AzureService**

Azure metadata service provides the metadata for Microsoft Azure cloud platform.

Azure metadata is offered via multiple sources like HTTP metadata, config-drive metadata and KVP (Hyper-V Key-Value Pair Data Exchange). This implementation uses only HTTP and config-drive metadata sources.

Azure service implements the interface to notify the cloud provider when the instance has started provisioning, completed provisioning and if the provisioning failed.

Metadata version used: *2015-04-05*.

Capabilities:

- instance id
- hostname
- public keys
- WinRM authentication certificates
- admin user name
- admin user password
- user data
- post RDP certificate thumbprint
- provisioning status
- Windows Update status
- VM agent configuration
- licensing configuration
- ephemeral disk warning

Config options for *azure* section:

- transport_cert_store_name (string: Windows Azure Environment")

Config options for *default* section:

- retry_count (integer: 5)
- retry_count_interval (integer: 4)

3.12 Empty Metadata Service

class cloudbaseinit.metadata.services.base.**EmptyMetadataService**

The empty metadata service can be used to run plugins that do not rely on metadata service information, like setting NTP, MTU, extending volumes, local scripts execution, licensing, etc.

It can be used also as a fallback metadata service, in case no other previous metadata service could be loaded.

EmptyMetadataService does not support the following plugins:

- cloudbaseinit.plugins.windows.createuser.CreateUserPlugin
- cloudbaseinit.plugins.common.setuserpassword.SetUserPasswordPlugin
- cloudbaseinit.plugins.common.sshpublickeys.SetUserSSHPublicKeysPlugin
- cloudbaseinit.plugins.windows.winrmcertificateauth.ConfigWinRMCertificateAuthPlugin

If any of the plugins defined above are executed, they will fail with exception NotExistingMetadataException. The reason for the hardcoded failure is that these plugins rely on metadata to execute correctly. If metadata like username or password is not provided, these plugins can lock or misconfigure the user, leading to unwanted problems.

Note: If a service returns an empty instance-id (like EmptyMetadataService does), all the plugins will be executed at every cloudbase-init run (reboot, service restart). Plugins that set NTP, MTU, extend volumes are idempotent and can be re-executed with no issues. Make sure that if you configure cloudbase-init to run local scripts, those local scripts are idempotent.

3.13 VMware GuestInfo Service

class cloudbaseinit.metadata.services.vmwareguestinfoservice.**VMwareGuestInfoService**

VMwareGuestInfoService is a metadata service which uses VMware's rpctool to extract guest metadata and userdata configured for machines running on VMware hypervisors.

The VMware RPC tool used to query the instance metadata and userdata needs to be present at the config option path.

Both json and yaml are supported as metadata formats. The metadata / userdata can be encoded in base64, gzip or gzip+base64.

Example metadata in yaml format:

```
instance-id: cloud-vm
local-hostname: cloud-vm
admin-username: cloud-username
admin-password: Passw0rd
public-keys-data: |
```

(continues on next page)

(continued from previous page)

```
  ssh-key 1
  ssh-key 2
```

This metadata content needs to be set as string in the guestinfo dictionary, thus needs to be converted to base64 (it is recommended to gzip it too). To convert to gzip+base64 format:

```
cat metadata.yml | gzip.exe -9 | base64.exe -w0
```

The output of the gzip+base64 conversion needs to be set in the instance guestinfo, along with the encoding of the metadata / userdata.

For more information on how to achieve this, please check https://github.com/vmware/cloud-init-vmware-guestinfo#configuration

This is an example how to set the information from the instance:

```
<rpctool_path> "info-set guestinfo.metadata <gzip+base64-encoded-metadata>"
<rpctool_path> "info-set guestinfo.metadata.encoding gzip+base64"
<rpctool_path> "info-set guestinfo.userdata <gzip+base64-encoded-userdata>"
<rpctool_path> "info-set guestinfo.userdata.encoding gzip+base64"
```

Capabilities:

- instance id
- hostname
- public keys
- admin user name
- admin user password
- user data

Config options for *vmwareguestinfo* section:

- vmware_rpctool_path (string: "%ProgramFiles%/VMware/VMware Tools/rpctool.exe")

3.14 Google Compute Engine Service

class cloudbaseinit.metadata.services.gceservice.**GCEService**

GCE metadata service provides the metadata for instances running on Google Compute Engine.

GCE metadata is offered via an internal HTTP metadata endpoint, reachable at the magic URL *http://metadata.google.internal/computeMetadata/v1/*. More information can be found in the GCE metadata documents.

To provide userdata to be executed by the instance (in cloud-config format, for example), use the user-data and user-data-encoding instance metadata keys.

Capabilities:

- instance id
- hostname
- public keys
- user data

Config options for *gce* section:

- metadata_base_url (string: http://metadata.google.internal/computeMetadata/v1/")
- https_allow_insecure (bool: False)
- https_ca_bundle (string: None)

Config options for *default* section:

- retry_count (integer: 5)
- retry_count_interval (integer: 4)

CHAPTER 4

Plugins

Plugins execute actions based on the metadata obtained by the loaded service. They are intended to configure the instance using data provided by the underlying cloud and by the user who created the instance. There are three stages for the plugins' execution:

1. The **PRE_NETWORKING** stage (for setting up the network, before doing any valid web request.

2. The **PRE_METADATA_DISCOVERY** stage (additional configuration before the metadata service discovery).

3. The default **MAIN** stage, which holds the rest of the plugins which are (re)executed according to their saved status. The metadata service loaded needs to provide an instance id for the plugins to be able to have a saved status. If the metadata service cannot provide an instance id, the plugins state from the **MAIN** stage cannot be saved, and therefore, all plugins will be executed at every boot.

Note that the plugins from the two stages are executed each time the cloudbase-init service starts (those plugins do not have saved status).

Just before the **MAIN** stage, the metadata service can report to the cloud service that the provision started. After the **MAIN** stage ended, the metadata service can report to the cloud service that the provisioning completed successfully or failed.

4.1 Configuring selected plugins

By default, only a subset of plugins is executed. The plugins are:

```
[DEFAULT]
plugins = cloudbaseinit.plugins.common.mtu.MTUPlugin, cloudbaseinit.plugins.windows.
↪ntpclient.NTPClientPlugin, cloudbaseinit.plugins.common.sethostname.
↪SetHostNamePlugin, cloudbaseinit.plugins.windows.createuser.CreateUserPlugin,␣
↪cloudbaseinit.plugins.common.networkconfig.NetworkConfigPlugin, cloudbaseinit.
↪plugins.windows.licensing.WindowsLicensingPlugin, cloudbaseinit.plugins.common.
↪sshpublickeys.SetUserSSHPublicKeysPlugin, cloudbaseinit.plugins.windows.
↪extendvolumes.ExtendVolumesPlugin, cloudbaseinit.plugins.common.userdata.
↪UserDataPlugin, cloudbaseinit.plugins.common.setuserpassword.SetUserPasswordPlugin,␣
↪cloudbaseinit.plugins.windows.winrmlistener.ConfigWinRMListenerPlugin,␣
↪cloudbaseinit.plugins.windows.winrmcertificateauth.ConfigWinRMCertificateAuthPlugin,
↪ cloudbaseinit.plugins.common.localscripts.LocalScriptsPlugin
```

(continues on next page)

(continued from previous page)

A custom list of plugins can be specified through the *plugins* option in the configuration file.

For more details on doing this, see *configuration* file in *Tutorial*.

4.2 Setting hostname (MAIN)

class cloudbaseinit.plugins.common.sethostname.**SetHostNamePlugin**

Sets the instance hostname. The hostname gets truncated to 15 characters for Netbios compatibility reasons if *netbios_host_name_compatibility* is set.

Config options:

- netbios_host_name_compatibility (bool: True)

Notes:

- Requires support in the metadata service.
- May require a system restart.

4.3 Creating user (MAIN)

class cloudbaseinit.plugins.windows.createuser.**CreateUserPlugin**

Creates (or updates if existing) a new user and adds it to a set of provided local groups. By default, it creates the user "Admin" under "Administrators" group, but this can be changed in the configuration file.

A random user password is set for the user. The password length is by default set to 20 and can be customized using the *user_password_length* configuration option.

If *rename_admin_user* is set to *True*, the user *Administrator* is renamed to the *username* config value or to the metadata service provided value.

Config options:

- username (string: "Admin")
- groups (list of strings: ["Administrators"])
- user_password_length (int: 20)
- rename_admin_user (bool: false)

Notes:

- The metadata service can provide the username. If the metadata service provides the admin username, it will override the *username* configuration value.

4.4 Setting password (MAIN)

class cloudbaseinit.plugins.common.setuserpassword.**SetUserPasswordPlugin**

Sets the cloud user's password. If a password has been provided in the metadata during boot it will be used, otherwise a random password will be generated, encrypted with the user's SSH public key and posted to the metadata provider.

An option called *inject_user_password* is set *True* by default to make available the use of metadata password which is found under the "admin_pass" field or through an URL request. If the option is set to *False* or if the password isn't found in metadata, then an attempt of using an already set password is done (usually a random value by the CreateUserPlugin plugin). With *first_logon_behaviour* you can control what happens with the password at the next logon. If this option is set to "always", the user will be forced to change the password at the next logon.

If it is set to "clear_text_injected_only", the user will be forced to change the password only if the password is a clear text password, coming from the metadata. The last option is "no", when the user is never forced to change the password.

Config options:

- username (string: "Admin")
- inject_user_password (bool: True)
- first_logon_behaviour (string: "clear_text_injected_only")
- user_password_length (int: 20)

Notes:

- The metadata service may provide the username. If the metadata service provides the admin username, it will override the *username* configuration value.
- May run at every boot to (re)set and post the password if the metadata service supports this behaviour.

4.5 Static networking (MAIN)

class cloudbaseinit.plugins.common.networkconfig.**NetworkConfigPlugin**

Statically configures each network adapter for which corresponding details are found into metadata. The details/addresses association is done using MAC matching and if this fails, then name or interface index matching. The basic setting is based on IPv4 addresses, but it supports IPv6 addresses too if they are enabled and exposed to the metadata. The purpose of this plugin is to configure network adapters, for which the DHCP server is disabled, to have internet access and static IPs.

NIC teaming (bonding) is supported and uses NetLBFO implementation.

Notes:

- Requires support in the metadata service.
- May require a system restart.

4.6 Saving public keys (MAIN)

class cloudbaseinit.plugins.common.sshpublickeys.**SetUserSSHPublicKeysPlugin**

Creates an **authorized_keys** file in the user's home directory containing the SSH keys provided in the metadata. It is needed by the plugin responsible for encrypting and setting passwords.

Config options:

- username (string: "Admin")

Notes:

- Requires support in the metadata service. The metadata service provides the SSH public keys.
- The metadata service can provide the username. If the metadata service provides the admin username, it will override the *username* configuration value.

4.7 Volume expanding (MAIN)

class cloudbaseinit.plugins.windows.extendvolumes.**ExtendVolumesPlugin**

Extends automatically a disk partition to its maximum size. This is useful when booting images with different flavors. By default, all the volumes are extended, but you can select specific ones by populating with their indexes the *volumes_to_extend* option.

Config options:

- volumes_to_extend (list of integers: None)

Notes:

- Runs at every boot.

4.8 WinRM listener (MAIN)

class cloudbaseinit.plugins.windows.winrmlistener.**ConfigWinRMListenerPlugin**

Configures a WinRM HTTPS listener to allow remote management via WinRM or PowerShell.

If *winrm_enable_basic_auth* is set to True, it enables basic authentication (authentication using username and password) for the WinRM listeners.

If *winrm_configure_http_listener* is set to True, the WinRM http listener will also be enabled.

Config options:

- winrm_enable_basic_auth (bool: True)
- winrm_configure_https_listener (bool: True)
- winrm_configure_http_listener (bool: False)

Notes:

- The metadata service can provide the listeners configuration (protocol and certificate thumbprint).
- May run at every boot. If the *WinRM* Windows service does not exist, it will run at the next boot.

4.9 WinRM certificate (MAIN)

class cloudbaseinit.plugins.windows.winrmcertificateauth.**ConfigWinRMCertificateAuthPlugin**

Enables password-less authentication for remote management via WinRS or PowerShell. Usually uses x509 embedded with UPN certificates.

Config options:

- username (string: "Admin")

Notes

- Requires support in the metadata service. The metadata service must provide the certificate metadata. The admin user password needs to be present, either from the metadata, either as shared data set by running CreateUserPlugin or SetUserPasswordPlugin. The metadata service can provide the username. If the metadata service provides the admin username, it will override the *username* configuration value.
- How to use this feature: http://www.cloudbase.it/windows-without-passwords-in-openstack/

4.10 Local Scripts execution (MAIN)

class cloudbaseinit.plugins.common.localscripts.**LocalScriptsPlugin**

Executes any script (powershell, batch, python etc.) located in the following path indicated by *local_scripts_path* option.

More details about the supported scripts and content can be found in *Tutorial* on *file execution* subject.

Config options:

- local_scripts_path (string: None)

Notes:

- May require a system restart.
- May run at every boot. It depends on the exit codes of the scripts.

4.11 Licensing (MAIN)

class cloudbaseinit.plugins.windows.licensing.**WindowsLicensingPlugin**

Activates the Windows instance if the *activate_windows* option is *True*. If *set_kms_product_key* or *set_avma_product_key* are set, it will use that KMS or AVMA product key in Windows.

If *kms_host* is set, it will set the provided host as the KMS licensing server.

Config options:

- activate_windows (bool: False)
- set_kms_product_key (bool: False)
- set_avma_product_key (bool: False)
- kms_host (string: None)
- log_licensing_info (bool: True)

Notes:

- The metadata service can provide the KMS host, overriding the configuration option *kms_host*. The metadata service can provide the avma_product_key, overriding the configuration option *set_avma_product_key*.

4.12 Clock synchronization (PRE_NETWORKING)

class cloudbaseinit.plugins.windows.ntpclient.**NTPClientPlugin**

Applies NTP client info based on the DHCP server options, if available. This behavior is enabled only when the *ntp_use_dhcp_config* option is set to *True* (which by default is *False*).

If *real_time_clock_utc* is set to True, it will set the real time clock to use universal time. If set to *False*, it will set the real time clock to use the local time.

Config options:

- ntp_use_dhcp_config (bool: False)
- real_time_clock_utc (bool: False)
- ntp_enable_service (bool: True)

Notes:

- May require a reboot.
- May run at every boot.

4.13 MTU customization (PRE_METADATA_DISCOVERY)

class cloudbaseinit.plugins.common.mtu.**MTUPlugin**

Sets the network interfaces MTU based on the value provided by the DHCP server options, if available and enabled (by default is *True*). This is particularly useful for cases in which a lower MTU value is required for networking (e.g. OpenStack GRE Neutron Open vSwitch configurations).

Config options:

- mtu_use_dhcp_config (bool: True)

Notes:

- Runs at every boot.

4.14 User data (MAIN)

class cloudbaseinit.plugins.common.userdata.**UserDataPlugin**

Executes custom scripts provided by user data metadata as plain text or compressed with Gzip. More details, examples and possible formats here: *Userdata*.

4.15 Trim Config (MAIN)

class cloudbaseinit.plugins.common.trim.**TrimConfigPlugin**

Enables or disables TRIM delete notifications for the underlying storage device.

Config options:

- trim_enabled (bool: False)

4.16 San Policy Config (MAIN)

class cloudbaseinit.plugins.windows.sanpolicy.**SANPolicyPlugin**

If not None, the SAN policy is set to the given value of the configuration option *san_policy*. The possible values are: OnlineAll, OfflineAll or OfflineShared.

Config options:

- san_policy (string: None)

4.17 RDP Settings Config (MAIN)

class cloudbaseinit.plugins.windows.rdp.**RDPSettingsPlugin**

Sets the registry key *KeepAliveEnable*, to enable or disable the RDP keep alive functionality.

Config options:

- rdp_set_keepalive (bool: False)

4.18 RDP Post Certificate Thumbprint (MAIN)

class cloudbaseinit.plugins.windows.rdp.**RDPPostCertificateThumbprintPlugin**

Posts the RDP certificate thumbprint to the metadata service endpoint.

Notes:

- Requires support in the metadata service. The metadata service should expose an HTTP endpoint where the certificate thumbprint can be posted.

4.19 Page Files (MAIN)

class cloudbaseinit.plugins.windows.pagefiles.**PageFilesPlugin**

Sets custom page files according to the config options.

Config options:

- page_file_volume_labels (array: [])
- page_file_volume_mount_points (array: [])

Notes:

- May require a reboot. If the page file is configured, a reboot is required.
- Runs at every boot.

4.20 Display Idle Timeout Config (MAIN)

class cloudbaseinit.plugins.windows.displayidletimeout.**DisplayIdleTimeoutConfigPlugin**

Sets the idle timeout, in seconds, before powering off the display. Set 0 to leave the display always on.

Config options:

- display_idle_timeout (int: 0)

4.21 Boot Status Policy Config (MAIN)

class cloudbaseinit.plugins.windows.bootconfig.**BootStatusPolicyPlugin**

Sets the Windows BCD boot status policy according to the config option. When set, the only possible value for *bcd_boot_status_policy* is *ignoreallfailures*.

Config options:

- bcd_boot_status_policy (string: None)

4.22 BCD Config (MAIN)

class cloudbaseinit.plugins.windows.bootconfig.**BCDConfigPlugin**

A unique disk ID is needed to avoid disk signature collisions. This plugin resets the boot disk id and enables auto-recovery in the BCD store.

Config options:

- set_unique_boot_disk_id (bool: False)
- bcd_enable_auto_recovery (bool: False)

4.23 Ephemeral Disk Config (MAIN)

class cloudbaseinit.plugins.common.ephemeraldisk.**EphemeralDiskPlugin**

Sets the ephemeral disk data loss warning file. On public clouds like Azure, the ephemeral disk should contain a read only file with data loss warning text, that warns the user to not use the ephemeral disk as a persistent storage disk.

Config options:

- ephemeral_disk_volume_label (string: None)
- ephemeral_disk_volume_mount_point (string: None)
- ephemeral_disk_data_loss_warning_path (string: None)

Notes:

- Requires support in the metadata service. The metadata service should provide the disk data loss warning text.

4.24 Windows Auto Updates (MAIN)

class cloudbaseinit.plugins.windows.updates.**WindowsAutoUpdatesPlugin**

Enables automatic Windows updates based on the user configuration or the metadata service information. The metadata service setting takes priority over the configuration option.

Config options:

- enable_automatic_updates (bool: False)

Notes:

- If the metadata service provides the information needed to enable the automatic updates, it will override the *enable_automatic_updates* configuration value.

4.25 Server Certificates (MAIN)

class cloudbaseinit.plugins.windows.certificates.**ServerCertificatesPlugin**

Imports X509 certificates into the desired store location. The metadata service provides the certificate and key in a PFX archive, their store location and store name.

Notes:

- Requires support in the metadata service.

4.26 Azure Guest Agent (MAIN)

class cloudbaseinit.plugins.windows.azureguestagent.**AzureGuestAgentPlugin**

Installs Azure Guest agent, which is required for the Azure cloud platform.

Notes:

- Requires support in the metadata service. Azure metadata service should provide the agent package provisioning data.

CHAPTER 5

Userdata

The *userdata* is the user custom content exposed to the guest instance by the currently deployed and running cloud infrastructure. Its purpose is to provide additional data for the instance to customize it as much as you need, if the cloud initialization service does support this feature.

Fortunately, *cloudbase-init* is able to interpret and use this kind of user specific data in multiple ways. In most of the cases, the thing that indicates of what type is the processed data is usually the **first line**.

Currently supported contents:

5.1 PEM certificate

—–BEGIN CERTIFICATE—–

This one should start with a PEM specific beginning header, which will be eventually parsed by the *configuration drive* and *web API* OpenStack services and used by the *WinRM certificate (MAIN)* plugin for storing and using it.

5.2 Batch

rem cmd

The file is executed in a *cmd.exe* shell (can be changed with the *COMSPEC* environment variable).

5.3 PowerShell

#ps1 or **#ps1_sysnative** (system native)

#ps1_x86 (Windows On Windows 32bit)

Execute PowerShell scripts using the desired executable. For finding out more about the system nativeness thing, click *here*.

5.4 Bash

#!/bin/bash

A bash shell needs to be installed in the system and available in the *PATH* in order to use this feature.

5.5 Python

#!/usr/bin/env python

Python is available by default with the build itself, but also it must be in the system *PATH*.

5.6 EC2 format

There is no "first line" here, but the content should follow a XML pattern with valid Batch/PowerShell script contents under **script** or **powershell** enclosing tags like in this example:

```
<script>
set root=%SystemDrive%
echo ec2dir>%root%\ec2file.txt
</script>

<powershell>
$root = $env:SystemDrive
$dname = Get-Content "$root\ec2file.txt"
New-Item -path "$root\$dname" -type directory
</powershell>
```

Note: http://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/UsingConfig_WinAMI.html

5.7 Cloud config

#cloud-config

Cloud-config YAML configuration as supported by *cloud-init*, excluding Linux specific content. The following cloud-config directives are supported:

- write_files - Defines a set of files which will be created on the local filesystem. It can be a list of items or only one item, with the following attributes:
    1. path - Absolute path on disk where the content should be written.
    2. content - The content which will be written in the given file.
    3. permissions - Integer representing file permissions.
    4. encoding - The encoding of the data in content. Supported encodings are: b64, base64 for base64-encoded content, gz, gzip for gzip encoded content, gz+b64, gz+base64, gzip+b64, gzip+base64 for base64 encoded gzip content.

  Examples:

```
#cloud-config
write_files:
   encoding: b64
   content: NDI=
   path: C:\test
   permissions: '0o466'
```

```
#cloud-config
write_files:
   -   encoding: b64
       content: NDI=
       path: C:\b64
       permissions: '0644'
   -   encoding: base64
       content: NDI=
       path: C:\b64_1
       permissions: '0644'
   -   encoding: gzip
       content: !!binary |
           H4sIAGUfoFQC/zMxAgCIsCQyAgAAAA==
       path: C:\gzip
       permissions: '0644'
```

- set_timezone - Change the underlying timezone.

  Example:

```
#cloud-config
set_timezone: Asia/Tbilisi
```

- set_hostname - Override the already default set hostname value (taken from metadata).

  If the hostname is changed, a reboot will be required.

  Example:

```
#cloud-config
set_hostname: newhostname
```

- groups - Create local groups and add existing users to those local groups.

  The definition of the groups consists of a list in the format:

  <group_name>: [<user1>, <user2>]

  The list of users can be empty, when creating a group without members.

  Example:

```
groups:
  - windows-group: [user1, user2]
  - cloud-users
```

- users - Create and configure local users.

  The users are defined as a list. Each element from the list represents a user. Each user can have the the following attributes defined:

  1. name - The username (required string).
  2. gecos - the user description.

3. primary_group - the user's primary group.
4. groups - the user's groups. On Windows, primary_group and groups are concatenated.
5. passwd - the user's password. On Linux, the password is a hashed string, whereas on Windows the password is a plaintext string. If the password is not defined, a random password will be set.
6. inactive - boolean value, defaults to False. If set to True, the user will be disabled.
7. expiredate - a string in the format <year>-<month>-<day>. Example: 2020-10-01.
8. ssh_authorized_keys - a list of SSH public keys, that will be set in ~/.ssh/authorized_keys.

Example:

```
users:
  -
    name: Admin
  -
    name: brian
    gecos: 'Brian Cohen'
    primary_group: Users
    groups: cloud-users
    passwd: StrongPassw0rd
    inactive: False
    expiredate: 2020-10-01
    ssh_authorized_keys:
      - ssh-rsa AAAB...byV
      - ssh-rsa AAAB...ctV
```

- ntp - Set NTP servers. The definition is a dict with the following attributes:

  1. enabled - Boolean value, defaults to True, to enable or disable the NTP config.
  2. servers - A list of NTP servers.
  3. pools - A list of NTP pools.

  The servers and pools are aggregated, servers being the first ones in the list. On Windows, there is no difference between an NTP pool or server.

  Example:

```
#cloud-config
ntp:
  enabled: True
  servers: ['my.ntp.server.local', '192.168.23.2']
  pools: ['0.company.pool.ntp.org', '1.company.pool.ntp.org']
```

- runcmd - Directive that can contain a list of commands that will be executed, in the order of their definition.

  A command can be defined as a string or as a list of strings, the first one being the executable path.

  On Windows, the commands are aggregated into a file and executed with *cmd.exe*. The userdata exit codes can be used to request a reboot: *File execution*.

  Example:

```
#cloud-config
runcmd:
  - 'dir C:\\'
  - ['echo', '1']
```

The cloud-config directives are executed by default in the following order: write_files, set_timezone, set_hostname, ntp, groups, users, runcmd. Use config option *cloud_config_plugins* to filter or to change the order of the cloud config plugins.

The execution of set_hostname or runcmd can request a reboot if needed. The reboot is performed at the end of the cloud-config execution (after all the directives have been executed).

5.8 Multi-part content

MIME multi-part user data is supported. The content will be handled based on the content type.

- text/x-shellscript - Any script to be executed: PowerShell, Batch, Bash or Python.
- text/part-handler - A script that can manage other content type parts. This is used in particular by Heat / CFN templates, although Linux specific.
- text/x-cfninitdata - Heat / CFN content. Written to the path provided by *heat_config_dir* option which defaults to "C:\cfn". (examples of Heat Windows templates)

5.9 Sysnativeness

When deciding which path to use for system executable files...

On 32bit OSes, the return value will be the *System32* directory, which contains 32bit programs. On 64bit OSes, the return value may be different, depending on the Python bits and the *sysnative* parameter. If the Python interpreter is 32bit, the return value will be *System32* (containing 32bit programs) if *sysnative* is set to False and *Sysnative* otherwise. But if the Python interpreter is 64bit and *sysnative* is False, the return value will be *SysWOW64* and *System32* for a True value of *sysnative*.

Why this behavior and what is the purpose of *sysnative* parameter?

On a 32bit OS the things are clear, there is one *System32* directory containing 32bit applications and that's all. On a 64bit OS, there's a *System32* directory containing 64bit applications and a compatibility one named *SysWOW64* (WindowsOnWindows) containing the 32bit version of them. Depending on the Python interpreter's bits, the *sysnative* flag will try to bring the appropriate version of the system directory, more exactly, the physical *System32* or *SysWOW64* found on disk. On a WOW case (32bit interpreter on 64bit OS), a return value of *System32* will point to the physical *SysWOW64* directory and a return value of *Sysnative*, which is consolidated by the existence of this alias, will point to the real physical *System32* directory found on disk. If the OS is still 64bit and there is no WOW case (that means the interpreter is 64bit), the system native concept is out of discussion and each return value will point to the physical location it intends to.

On a 32bit OS the *sysnative* parameter has no meaning, but on a 64bit one, based on its value, it will provide a real/alias path pointing to system native applications if set to True (64bit programs) and to system compatibility applications if set to False (32bit programs). Its purpose is to provide the correct system paths by taking into account the Python interpreter bits too, because on a 32bit interpreter version, *System32* is not the same with the *System32* on a 64bit interpreter. Also, using a 64bit interpreter, the *Sysnative* alias will not work, but the *sysnative* parameter will take care to return *SysWOW64* if you explicitly want 32bit applications, by setting it to False.

CHAPTER 6

Configuration options reference

The following is an overview of all available configuration options in cloudbase-init.

6.1 DEFAULT

debug

Type boolean

Default `False`

Mutable This option can be changed without restarting.

If set to true, the logging level will be set to DEBUG instead of the default INFO level.

log_config_append

Type string

Default `<None>`

Mutable This option can be changed without restarting.

The name of a logging configuration file. This file is appended to any existing logging configuration files. For details about logging configuration files, see the Python logging module documentation. Note that when logging configuration files are used then all logging configuration is set in the configuration file and other logging configuration options are ignored (for example, log-date-format).

Table 1: Deprecated Variations

Group	Name
DEFAULT	log-config
DEFAULT	log_config

log_date_format

Type string

Default `%Y-%m-%d %H:%M:%S`

Defines the format string for %(asctime)s in log records. Default: the value above . This option is ignored if log_config_append is set.

log_file

Type string

Default `<None>`

(Optional) Name of log file to send logging output to. If no default is set, logging will go to stderr as defined by use_stderr. This option is ignored if log_config_append is set.

Table 2: Deprecated Variations

Group	Name
DEFAULT	logfile

log_dir

Type string

Default `<None>`

(Optional) The base directory used for relative log_file paths. This option is ignored if log_config_append is set.

Table 3: Deprecated Variations

Group	Name
DEFAULT	logdir

watch_log_file

Type boolean

Default `False`

Uses logging handler designed to watch file system. When log file is moved or removed this handler will open a new log file with specified path instantaneously. It makes sense only if log_file option is specified and Linux platform is used. This option is ignored if log_config_append is set.

use_syslog

Type boolean

Default `False`

Use syslog for logging. Existing syslog format is DEPRECATED and will be changed later to honor RFC5424. This option is ignored if log_config_append is set.

use_journal

Type boolean

Default `False`

Enable journald for logging. If running in a systemd environment you may wish to enable journal support. Doing so will use the journal native protocol which includes structured metadata in addition to log messages.This option is ignored if log_config_append is set.

syslog_log_facility

Type string

Default `LOG_USER`

Syslog facility to receive log lines. This option is ignored if log_config_append is set.

use_json

Type boolean

Default `False`

Use JSON formatting for logging. This option is ignored if log_config_append is set.

use_stderr

Type boolean

Default `False`

Log output to standard error. This option is ignored if log_config_append is set.

use_eventlog

Type boolean

Default `False`

Log output to Windows Event Log.

log_rotate_interval

Type integer

Default `1`

The amount of time before the log files are rotated. This option is ignored unless log_rotation_type is setto "interval".

log_rotate_interval_type

Type string

Default `days`

Valid Values Seconds, Minutes, Hours, Days, Weekday, Midnight

Rotation interval type. The time of the last file change (or the time when the service was started) is used when scheduling the next rotation.

max_logfile_count

Type integer

Default `30`

Maximum number of rotated log files.

max_logfile_size_mb

Type integer

Default `200`

Log file maximum size in MB. This option is ignored if "log_rotation_type" is not set to "size".

log_rotation_type

Type string

Default `none`

Valid Values interval, size, none

Log rotation type.

Possible values

interval Rotate logs at predefined time intervals.

size Rotate logs once they reach a predefined size.

none Do not rotate log files.

logging_context_format_string

Type string

Default `%(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s`

Format string to use for log messages with context. Used by oslo_log.formatters.ContextFormatter

logging_default_format_string

Type string

Default `%(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s`

Format string to use for log messages when context is undefined. Used by oslo_log.formatters.ContextFormatter

logging_debug_format_suffix

Type string

Default `%(funcName)s %(pathname)s:%(lineno)d`

Additional data to append to log message when logging level for the message is DEBUG. Used by oslo_log.formatters.ContextFormatter

logging_exception_prefix

Type string

Default `%(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s`

Prefix each line of exception output with this format. Used by oslo_log.formatters.ContextFormatter

logging_user_identity_format

Type string

Default `%(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s`

Defines the format string for %(user_identity)s that is used in logging_context_format_string. Used by oslo_log.formatters.ContextFormatter

default_log_levels

Type list

Default `['amqp=WARN', 'amqplib=WARN', 'boto=WARN', 'qpid=WARN', 'sqlalchemy=WARN', 'suds=INFO', 'oslo.messaging=INFO', 'oslo_messaging=INFO', 'iso8601=WARN', 'requests.packages.urllib3.connectionpool=WARN', 'urllib3.connectionpool=WARN', 'websocket=WARN', 'requests.packages.urllib3.util.retry=WARN',`

```
'urllib3.util.retry=WARN', 'keystonemiddleware=WARN',
'routes.middleware=WARN', 'stevedore=WARN', 'taskflow=WARN',
'keystoneauth=WARN', 'oslo.cache=INFO', 'oslo_policy=INFO',
'dogpile.core.dogpile=INFO']
```

List of package logging levels in logger=LEVEL pairs. This option is ignored if log_config_append is set.

publish_errors

Type boolean

Default `False`

Enables or disables publication of error events.

instance_format

Type string

Default `"[instance: %(uuid)s] "`

The format for an instance that is passed with the log message.

instance_uuid_format

Type string

Default `"[instance: %(uuid)s] "`

The format for an instance UUID that is passed with the log message.

rate_limit_interval

Type integer

Default `0`

Interval, number of seconds, of log rate limiting.

rate_limit_burst

Type integer

Default `0`

Maximum number of logged messages per rate_limit_interval.

rate_limit_except_level

Type string

Default `CRITICAL`

Log level name used by rate limiting: CRITICAL, ERROR, INFO, WARNING, DEBUG or empty string. Logs with level greater or equal to rate_limit_except_level are not filtered. An empty string means that all levels are filtered.

fatal_deprecations

Type boolean

Default `False`

Enables or disables fatal status of deprecations.

allow_reboot

Type boolean

Default `True`

Allows OS reboots requested by plugins

stop_service_on_exit

Type boolean

Default `True`

In case of execution as a service, specifies if the service must be gracefully stopped before exiting

check_latest_version

Type boolean

Default `False`

Check if there is a newer version of cloudbase-init available. If this option is activated, a log message will be emitted if there is a newer version available.

retry_count

Type integer

Default `5`

Max. number of attempts for fetching metadata in case of transient errors

retry_count_interval

Type floating point

Default `4`

Interval between attempts in case of transient errors, expressed in seconds

mtools_path

Type string

Default `<None>`

Path to "mtools" program suite, used for interacting with VFAT filesystems

bsdtar_path

Type string

Default `bsdtar.exe`

Path to "bsdtar", used to extract ISO ConfigDrive files

netbios_host_name_compatibility

Type boolean

Default `True`

Truncates the hostname to 15 characters for Netbios compatibility

logging_serial_port_settings

Type string

Default `<None>`

Serial port logging settings. Format: "port,baudrate,parity,bytesize", e.g.: "COM1,115200,N,8". Set to None (default) to disable.

activate_windows

Type boolean

Default `False`

Activates Windows automatically

set_kms_product_key

Type boolean

Default `False`

Sets the KMS product key for this operating system

set_avma_product_key

Type boolean

Default `False`

Sets the AVMA product key for this operating system

kms_host

Type string

Default `<None>`

The KMS host address in form <host>[:<port>], e.g: "kmshost:1688"

log_licensing_info

Type boolean

Default `True`

Logs the operating system licensing information

winrm_enable_basic_auth

Type boolean

Default `True`

Enables basic authentication for the WinRM HTTPS listener

winrm_configure_http_listener

Type boolean

Default `False`

Configures the WinRM HTTP listener

winrm_configure_https_listener

Type boolean

Default `True`

Configures the WinRM HTTPS listener

volumes_to_extend

Type list

Default `<None>`

List of volumes that need to be extended if contiguous space is available on the disk. By default all the available volumes can be extended. Volumes must be specified using a comma separated list of volume indexes, e.g.: "1,2"

san_policy

Type string

Default `<None>`

Valid Values OnlineAll, OfflineAll, OfflineShared

If not None, the SAN policy is set to the given value

local_scripts_path

Type string

Default `<None>`

Path location containing scripts to be executed when the plugin runs

mtu_use_dhcp_config

Type boolean

Default `True`

Configures the network interfaces MTU based on the values provided via DHCP

username

Type string

Default `Admin`

User to be added to the system or updated if already existing

groups

Type list

Default `['Administrators']`

List of local groups to which the user specified in "username" will be added

rename_admin_user

Type boolean

Default `False`

Renames the builtin admin user instead of creating a new user

heat_config_dir

Type string

Default `C:\cfn`

The directory where the Heat configuration files must be saved

ntp_enable_service

Type boolean

Default `True`

Enables the NTP client service

ntp_use_dhcp_config

Type boolean

Default `False`

Configures NTP client time synchronization using the NTP servers provided via DHCP

real_time_clock_utc

Type boolean

Default `False`

Sets the real time clock to use universal time (True) or local time (False)

inject_user_password

Type boolean

Default `True`

Set the password provided in the configuration. If False or no password is provided, a random one will be set

first_logon_behaviour

Type string

Default `clear_text_injected_only`

Valid Values clear_text_injected_only, no, always

Control the behaviour of what happens at next logon. If this option is set to *always*, then the user will be forced to change the password at next logon. If it is set to *clear_text_injected_only*, then the user will have to change the password only if the password is a clear text password, coming from the metadata. The last option is *no*, when the user is never forced to change the password.

metadata_services

Type list

Default
```
['cloudbaseinit.metadata.services.httpservice.
   HttpService', 'cloudbaseinit.metadata.services.configdrive.
   ConfigDriveService', 'cloudbaseinit.metadata.services.
   ec2service.EC2Service', 'cloudbaseinit.metadata.services.
   maasservice.MaaSHttpService', 'cloudbaseinit.metadata.services.
   cloudstack.CloudStack', 'cloudbaseinit.metadata.services.
   opennebulaservice.OpenNebulaService']
```

List of enabled metadata service classes, to be tested for availability in the provided order. The first available service will be used to retrieve metadata

plugins

Type list

Default
```
['cloudbaseinit.plugins.common.mtu.MTUPlugin',
   'cloudbaseinit.plugins.windows.ntpclient.NTPClientPlugin',
   'cloudbaseinit.plugins.common.sethostname.SetHostNamePlugin',
   'cloudbaseinit.plugins.windows.createuser.CreateUserPlugin',
   'cloudbaseinit.plugins.common.networkconfig.
   NetworkConfigPlugin', 'cloudbaseinit.plugins.windows.
   licensing.WindowsLicensingPlugin', 'cloudbaseinit.
   plugins.common.sshpublickeys.SetUserSSHPublicKeysPlugin',
   'cloudbaseinit.plugins.windows.extendvolumes.
   ExtendVolumesPlugin', 'cloudbaseinit.plugins.common.
   userdata.UserDataPlugin', 'cloudbaseinit.plugins.common.
   setuserpassword.SetUserPasswordPlugin', 'cloudbaseinit.
   plugins.windows.winrmlistener.ConfigWinRMListenerPlugin',
```

```
'cloudbaseinit.plugins.windows.winrmcertificateauth.
ConfigWinRMCertificateAuthPlugin', 'cloudbaseinit.plugins.
common.localscripts.LocalScriptsPlugin']
```

List of enabled plugin classes, to be executed in the provided order

user_data_plugins

Type list

Default `['cloudbaseinit.plugins.common.userdataplugins.
parthandler.PartHandlerPlugin', 'cloudbaseinit.plugins.
common.userdataplugins.cloudconfig.CloudConfigPlugin',
'cloudbaseinit.plugins.common.userdataplugins.cloudboothook.
CloudBootHookPlugin', 'cloudbaseinit.plugins.common.
userdataplugins.shellscript.ShellScriptPlugin',
'cloudbaseinit.plugins.common.userdataplugins.multipartmixed.
MultipartMixedPlugin', 'cloudbaseinit.plugins.common.
userdataplugins.heat.HeatPlugin']`

List of enabled userdata content plugins

cloud_config_plugins

Type list

Default `[]`

List which contains the name of the cloud config plugins ordered by priority.

rdp_set_keepalive

Type boolean

Default `True`

Sets the RDP KeepAlive policy

bcd_boot_status_policy

Type string

Default `<None>`

Valid Values ignoreallfailures

Sets the Windows BCD boot status policy

bcd_enable_auto_recovery

Type boolean

Default `False`

Enables or disables the BCD auto recovery

set_unique_boot_disk_id

Type boolean

Default `True`

Sets a new random unique id on the boot disk to avoid collisions

display_idle_timeout

Type integer

Default `0`

The idle timeout, in seconds, before powering off the display. Set 0 to leave the display always on

page_file_volume_labels

Type list

Default `[]`

Labels of volumes on which a Windows page file needs to be created. E.g.: "Temporary Storage"

page_file_volume_mount_points

Type list

Default `[]`

Volume mount points on which a Windows page file needs to be created. E.g.: "\?GLOBALROOTdeviceHarddisk1Partition1"

trim_enabled

Type boolean

Default `False`

Enables or disables TRIM delete notifications for the underlying storage device.

process_userdata

Type boolean

Default `True`

Processes the userdata content based on the type, e.g. executing a PowerShell script

userdata_save_path

Type string

Default `<None>`

Copies the userdata to the given file path. The path can include environment variables that will be expanded, e.g. "%SYSTEMDRIVE%CloudbaseInitUserData.bin"

enable_automatic_updates

Type boolean

Default `<None>`

If set, enables or disables automatic operating system updates.

metadata_report_provisioning_started

Type boolean

Default `False`

Reports to the metadata service that provisioning has started

metadata_report_provisioning_completed

Type boolean

Default `False`

Reports to the metadata service that provisioning completed successfully or failed

ephemeral_disk_volume_label

Type string

Default `<None>`

Ephemeral disk volume label, e.g.: "Temporary Storage"

ephemeral_disk_volume_mount_point

Type string

Default `<None>`

Ephemeral disk volume mount point, e.g.:"\?GLOBALROOTdeviceHarddisk1Partition1"

ephemeral_disk_data_loss_warning_path

Type string

Default `<None>`

Ephemeral disk data loss warning path, relative to the ephemeral disk volume path. E.g.: DATALOSS_WARNING_README.txt

user_password_length

Type integer

Default `20`

The length of the generated password for the user defined by the *username* config option.

6.2 azure

transport_cert_store_name

Type string

Default `Windows Azure Environment`

Certificate store name for metadata certificates

6.3 cloudstack

metadata_base_url

Type string

Default `http://10.1.1.1/`

The base URL where the service looks for metadata

Table 4: Deprecated Variations

Group	Name
DEFAULT	cloudstack_metadata_ip

password_server_port

Type integer

Default `8080`

The port number used by the Password Server.

https_allow_insecure

Type boolean

Default `False`

Whether to disable the validation of HTTPS certificates.

https_ca_bundle

Type string

Default `<None>`

The path to a CA_BUNDLE file or directory with certificates of trusted CAs.

add_metadata_private_ip_route

Type boolean

Default `False`

Add a route for the metadata ip address to the gateway

6.4 config_drive

raw_hdd

Type boolean

Default `True`

Look for an ISO config drive in raw HDDs

Table 5: Deprecated Variations

Group	Name
DEFAULT	config_drive_raw_hhd

Warning: This option is deprecated for removal. Its value may be silently ignored in the future.

cdrom

Type boolean

Default `True`

Look for a config drive in the attached cdrom drives

Table 6: Deprecated Variations

Group	Name
DEFAULT	config_drive_cdrom

Warning: This option is deprecated for removal. Its value may be silently ignored in the future.

vfat

Type boolean

Default `True`

Look for a config drive in VFAT filesystems

Table 7: Deprecated Variations

Group	Name
DEFAULT	config_drive_vfat

Warning: This option is deprecated for removal. Its value may be silently ignored in the future.

types

Type list

Default `['vfat', 'iso']`

Supported formats of a configuration drive

Table 8: Deprecated Variations

Group	Name
DEFAULT	config_drive_types

locations

Type list

Default `['partition', 'hdd', 'cdrom']`

Supported configuration drive locations

Table 9: Deprecated Variations

Group	Name
DEFAULT	config_drive_locations

6.5 ec2

metadata_base_url

Type string

Default `http://169.254.169.254/`

The base URL where the service looks for metadata

Table 10: Deprecated Variations

Group	Name
DEFAULT	ec2_metadata_base_url

add_metadata_private_ip_route

Type boolean

Default `True`

Add a route for the metadata ip address to the gateway

Table 11: Deprecated Variations

Group	Name
DEFAULT	ec2_add_metadata_private_ip_route

`https_allow_insecure`

Type boolean

Default `False`

Whether to disable the validation of HTTPS certificates.

`https_ca_bundle`

Type string

Default `<None>`

The path to a CA_BUNDLE file or directory with certificates of trusted CAs.

6.6 gce

`metadata_base_url`

Type string

Default `http://metadata.google.internal/computeMetadata/v1/`

The base URL where the service looks for metadata

`https_allow_insecure`

Type boolean

Default `False`

Whether to disable the validation of HTTPS certificates.

`https_ca_bundle`

Type string

Default `<None>`

The path to a CA_BUNDLE file or directory with certificates of trusted CAs.

6.7 maas

`metadata_base_url`

Type string

Default `<None>`

The base URL for MaaS metadata

Table 12: Deprecated Variations

Group	Name
DEFAULT	maas_metadata_url

oauth_consumer_key

Type string

Default ''

The MaaS OAuth consumer key

Table 13: Deprecated Variations

Group	Name
DEFAULT	maas_oauth_consumer_key

oauth_consumer_secret

Type string

Default ''

The MaaS OAuth consumer secret

Table 14: Deprecated Variations

Group	Name
DEFAULT	maas_oauth_consumer_secret

oauth_token_key

Type string

Default ''

The MaaS OAuth token key

Table 15: Deprecated Variations

Group	Name
DEFAULT	maas_oauth_token_key

oauth_token_secret

Type string

Default ''

The MaaS OAuth token secret

Table 16: Deprecated Variations

Group	Name
DEFAULT	maas_oauth_token_secret

https_allow_insecure

Type boolean

Default `False`

Whether to disable the validation of HTTPS certificates.

`https_ca_bundle`

Type string

Default `<None>`

The path to a CA_BUNDLE file or directory with certificates of trusted CAs.

6.8 openstack

`metadata_base_url`

Type string

Default `http://169.254.169.254/`

The base URL where the service looks for metadata

Table 17: Deprecated Variations

Group	Name
DEFAULT	metadata_base_url

`add_metadata_private_ip_route`

Type boolean

Default `True`

Add a route for the metadata ip address to the gateway

Table 18: Deprecated Variations

Group	Name
DEFAULT	add_metadata_private_ip_route

`https_allow_insecure`

Type boolean

Default `False`

Whether to disable the validation of HTTPS certificates.

`https_ca_bundle`

Type string

Default `<None>`

The path to a CA_BUNDLE file or directory with certificates of trusted CAs.

6.9 ovf

`config_file_name`

Type string

Default `ovf-env.xml`

Configuration file name

drive_label

Type string

Default `OVF ENV`

Look for configuration file in drives with this label

ns

Type string

Default `oe`

Namespace prefix for ovf environment

6.10 packet

metadata_base_url

Type string

Default `https://metadata.packet.net/`

The URL where the service looks for metadata

https_allow_insecure

Type boolean

Default `False`

Whether to disable the validation of HTTPS certificates.

https_ca_bundle

Type string

Default `<None>`

The path to a CA_BUNDLE file or directory with certificates of trusted CAs.

6.11 vmwareguestinfo

vmware_rpctool_path

Type string

Default `%ProgramFiles%/VMware/VMware Tools/rpctool.exe`

The local path where VMware rpctool is found

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Index

C

cloudbaseinit.metadata.services.azureservice.AzureService (*built-in class*), 15
cloudbaseinit.metadata.services.base.EmptyMetadataService (*built-in class*), 16
cloudbaseinit.metadata.services.cloudstack.CloudStack (*built-in class*), 12
cloudbaseinit.metadata.services.configdrive.ConfigDriveService (*built-in class*), 10
cloudbaseinit.metadata.services.ec2service.EC2Service (*built-in class*), 12
cloudbaseinit.metadata.services.gceservice.GCEService (*built-in class*), 17
cloudbaseinit.metadata.services.httpservice.HttpService (*built-in class*), 9
cloudbaseinit.metadata.services.maasservice.MaasHttpService (*built-in class*), 13
cloudbaseinit.metadata.services.nocloudservice.NoCloudConfigDriveService (*built-in class*), 11
cloudbaseinit.metadata.services.opennebulaservice.OpenNebulaService (*built-in class*), 13
cloudbaseinit.metadata.services.ovfservice.OvfService (*built-in class*), 14
cloudbaseinit.metadata.services.packet.PacketService (*built-in class*), 14
cloudbaseinit.metadata.services.vmwareguestinfoservice.VMwareGuestInfoService (*built-in class*), 16
cloudbaseinit.plugins.common.ephemeraldisk.EphemeralDiskPlugin (*built-in class*), 26
cloudbaseinit.plugins.common.localscripts.LocalScriptsPlugin (*built-in class*), 23
cloudbaseinit.plugins.common.mtu.MTUPlugin (*built-in class*), 24
cloudbaseinit.plugins.common.networkconfig.NetworkConfigPlugin (*built-in class*), 21
cloudbaseinit.plugins.common.sethostname.SetHostNamePlugin (*built-in class*), 20
cloudbaseinit.plugins.common.setuserpassword.SetUserPasswordPlugin (*built-in class*), 21
cloudbaseinit.plugins.common.sshpublickeys.SetUserS (*built-in class*), 21
cloudbaseinit.plugins.common.trim.TrimConfigPlugin (*built-in class*), 24
cloudbaseinit.plugins.common.userdata.UserDataPlugi (*built-in class*), 24
cloudbaseinit.plugins.windows.azureguestagent.Azure (*built-in class*), 27
cloudbaseinit.plugins.windows.bootconfig.BCDConfigP (*built-in class*), 26
cloudbaseinit.plugins.windows.bootconfig.BootStatus (*built-in class*), 26
cloudbaseinit.plugins.windows.certificates.ServerCe (*built-in class*), 27
cloudbaseinit.plugins.windows.createuser.CreateUser (*built-in class*), 20
cloudbaseinit.plugins.windows.displayidletimeout.Di (*built-in class*), 25
cloudbaseinit.plugins.windows.extendvolumes.ExtendV (*built-in class*), 22
cloudbaseinit.plugins.windows.licensing.WindowsLice (*built-in class*), 23
cloudbaseinit.plugins.windows.ntpclient.NTPClientPl (*built-in class*), 24
cloudbaseinit.plugins.windows.pagefiles.PageFilesPl (*built-in class*), 25
cloudbaseinit.plugins.windows.rdp.RDPPostCertificat (*built-in class*), 25
cloudbaseinit.plugins.windows.rdp.RDPSettingsPlugin (*built-in class*), 25
cloudbaseinit.plugins.windows.sanpolicy.SANPolicyPl (*built-in class*), 25
cloudbaseinit.plugins.windows.updates.WindowsAutoUp (*built-in class*), 26
cloudbaseinit.plugins.windows.winrmcertificateauth (*built-in class*), 22
cloudbaseinit.plugins.windows.winrmlistener.ConfigW (*built-in class*), 22