Autoinstall configuration reference manual

The autoinstall file uses the YAML format. At the top level is a single key, `autoinstall`, which contains a mapping of the keys described in this document. Unrecognized keys are ignored in version 1, but they will cause a fatal validation error in future versions.

Here is an example of a minimal autoinstall configuration:

```
autoinstall:
  version: 1
  identity:
   ...
```

At the top level is the `autoinstall` keyword. It contains a `version` section and an (incomplete) `identity` section, which are explained in more detail below. Any other key at the level of `autoinstall` results in an autoinstall validation error at run time.

> **Note**
>
> This behavior was first introduced during 24.04 (Noble). On any ISOs built before 24.04, you need to refresh the installer to see this behavior.
>
> Technically, in all but one case the top level `autoinstall` keyword is strictly unnecessary. This keyword is only necessary when serving autoinstall via cloud-config. For backwards compatibility, this format is still supported for delivery methods not based on cloud-config; however, it is **highly recommended** to use the format with a top-level `autoinstall` keyword because mistakes in this formatting are a common source of confusion.

Schema

Autoinstall configurations are validated against a JSON schema before they are used.

Command lists

Several configuration keys are lists of commands to be executed. Each command can be a string (in which case it is executed via `sh -c`) or a list, in which case it is executed directly. The commands are run as root. Any command exiting with a non-zero return code is considered an error and aborts the installation (except for error-commands, where it is ignored).

Top-level keys

The following keys can be used to configure various aspects of the installation. If the global `autoinstall` key is provided, then all "top-level keys" must be provided underneath it and "top-level" refers to this sub-level. The examples below demonstrate this structure.

> **Warning**
>
> In version 1, Subiquity emits warnings when encountering unrecognized keys. In later versions, it results in a fatal validation error, and the installation halts.

version

- **type:** integer
- **default:** no default

A future-proofing configuration file version field. Currently, this must be `1`.

interactive-sections

- **type:** list of strings
- **default:** []

A list of configuration keys to still show in the user interface (UI). For example:

```
autoinstall:
  version: 1
  interactive-sections:
    - network
  identity:
    username: ubuntu
    $crypted_pass
```

This example stops on the network screen and allows the user to change the defaults. If a value is provided for an interactive section, it is used as the default.

You can use the special section name of `*` to indicate that the installer should ask all the usual questions – in this case, the `autoinstall.yaml` file is an autoinstall file. It just provides a way to change the defaults in the UI.

Not all configuration keys correspond to screens in the UI. This documentation indicates if a given section can be interactive or not.

If there are any interactive sections at all, the reporting key is ignored.

early-commands

- **type:** command list
- **default:** no commands
- **can be interactive:** no

A list of shell commands to invoke as soon as the installer starts, in particular before probing for block and network devices. The autoinstall configuration is available at `/autoinstall.yaml` (irrespective of how it was provided), and the file is re-read after the `early-commands` have run to allow them to alter the configuration if necessary.

Example early commands:

```
autoinstall:
  # Pause the install just before starting to allow manual inspection/modifica
  # Unpause by creating the "/run/finish-early" file.
  early-commands:
    - while [ ! -f /run/finish-early ]; do sleep 1; done

autoinstall:
  # Replace the current autoinstall configuration with one provided by a trust
  early-commands:
    - wget -O /autoinstall.yaml $TRUSTED_SERVER_URL
```

locale

`.UTF-8`

- **can be interactive:** true

The locale to configure for the installed system.

locale examples:

```
autoinstall:
  # default behavior
  locale: "en_US.UTF-8"

autoinstall:
  # Greek locale
  locale: "el_GR"
```

refresh-installer

- **type:** mapping
- **default:** see below
- **can be interactive:** true

Controls whether the installer updates to a new version available in the given channel before continuing.

The mapping contains keys:

update

- **type:** boolean
- **default:** `false`

Whether to update or not.

channel

- **type:** string
- **default:** `"stable/ubuntu-$REL"`

The channel to check for updates.

Example:

```
autoinstall:
 # Refresh to the latest snap built from the "main" subiquity branch
 refresh-installer:
   update: true
   channel: latest/edge

autoinstall:
 # Refresh to the latest beta release
 refresh-installer:
   update: true
   channel: latest/beta
```

keyboard

- **type:** mapping, see below
- **default:** US English keyboard
- **can be interactive:** true

The layout (or layouts) of any attached keyboard. The mapping keys correspond to settings in the `/etc/default/keyboard` configuration file. See the *keyboard(5)* manual page for more details.

Multiple layouts may be configured using comma-separated values, as documented in *keyboard(5)*.

> **Note**
>
> Known issue: for TPM-backed encrypted installations with a passphrase, the disk can only be unlocked using the primary layout.

The mapping contains keys:

layout

- **type:** string
- **default:** `"us"`

Corresponds to the `XKBLAYOUT` setting.

variant

Corresponds to the `XKBVARIANT` setting.

toggle

- **type:** string or null
- **default:** `null`

Corresponds to the value of `grp:` option from the `XKBOPTIONS` setting. Acceptable values are (the installer does not validate these):

- `caps_toggle`
- `toggle`
- `rctrl_toggle`
- `rshift_toggle`
- `rwin_toggle`
- `menu_toggle`
- `alt_shift_toggle`
- `ctrl_shift_toggle`
- `ctrl_alt_toggle`
- `alt_caps_toggle`
- `lctrl_lshift_toggle`
- `lalt_toggle`
- `lctrl_toggle`
- `lshift_toggle`
- `lwin_toggle`
- `sclk_toggle`

keyboard examples:

```
autoinstall:
  # default behavior
  keyboard:
    layout: us
    variant: ""
    toggle: null

autoinstall:
  # use Alt and Shift to toggle between default US keyboard and
  # "Greek (simple)"
  keyboard:
    layout: "us,gr"
    variant: ",simple"
    toggle: alt_shift_toggle
```

source

- **type:** mapping, see below
- **default:** see below
- **can be interactive:** true

search_drivers

- **type:** boolean
- **default:** `true` (mostly, see below)

Whether the installer searches for available third-party drivers. When set to `false`, it disables the drivers screen and section.

The default is `true` for most installations, and `false` when a "core boot" or "enhanced secure boot" method is selected (where third-party drivers cannot be currently installed).

id

- **type:** string
- **default:** the default value as listed in install-sources

Identifier of the source to install (e.g., `ubuntu-server-minimal`). The correct ID to use is specific to the installation ISO. As this ID may change over time, the canonical place to find the correct value to use is the installation ISO itself, in the `casper/install-sources.yaml` file, in the `id`.

Current values:

- Ubuntu Server:
  - minimal: `ubuntu-server-minimal`
  - standard (default): `ubuntu-server`
- Ubuntu Desktop:
  - minimal (default): `ubuntu-desktop-minimal`
  - standard: `ubuntu-desktop`
- Ubuntu Budgie:
  - minimal: `ubuntu-budgie-desktop-minimal`
  - standard (default): `ubuntu-budgie-desktop`
- Ubuntu Cinnamon:
  - minimal: `ubuntucinnamon-desktop-minimal`
  - standard (default): `ubuntucinnamon-desktop`
- Edubuntu:
  - minimal: `edubuntu-desktop-minimal`
  - standard (default): `edubuntu-desktop`
- Ubuntu Kylin:
  - minimal: `ubuntukylin-desktop-minimal`
  - standard (default): `ubuntukylin-desktop`
- Ubuntu MATE:
  - minimal: `ubuntu-mate-desktop-minimal`
  - standard (default): `ubuntu-mate-desktop`
- Ubuntu Studio:
  - standard (default): `ubuntustudio-desktop`
- Xubuntu:
  - full ISO:
    - minimal: `xubuntu-desktop-minimal`
    - standard (default): `xubuntu-desktop`
  - minimal ISO:
    - minimal (default): `xubuntu-desktop-minimal`

source examples:

```
autoinstall:
  # default behavior
  source:
    search_drivers: true
    id: <the installation source marked as default in install-sources.yaml>

autoinstall:
  # on the Ubuntu Server ISO, install with the minimal source
  source:
    id: ubuntu-server-minimal

autoinstall:
  # on the Ubuntu Desktop ISO, install with the standard source
  source:
    id: ubuntu-desktop
```

network

- **type:** Netplan-format mapping, see below
- **default:** DHCP on interfaces named `eth*` or `en*`
- **can be interactive:** true

Netplan-formatted network configuration. This is applied during installation as well as in the installed system. The default is to interpret the configuration for the installation media, which runs DHCP version 4 on any interface with a name matching `eth*` or `en*` but then disables any interface that does not receive an address.

For example, to run DHCP version 6 on a specific network interface:

```
autoinstall:
  network:
    version: 2
    ethernets:
      enp0s31f6:
        dhcp6: true
```

proxy

- **type:** URL or `null`
- **default:** no proxy
- **can be interactive:** true

The proxy to configure both during installation and for `apt` and `snapd` in the target system. This setting is currently not honored when running the geoip lookup.

Example:

```
autoinstall:
  proxy: http://172.16.90.1:3128
```

apt

- **type:** mapping
- **default:** see below
- **can be interactive:** true

APT configuration, used both during the installation and once booted into the target system.

This section has historically used the same format as curtin, which is documented in the APT Source section of the curtin documentation. Nonetheless, some key differences with the format supported by curtin have been introduced:

- Subiquity supports an alternative format for the `primary` section, allowing configuration of a list of candidate primary mirrors. During installation, Subiquity automatically tests the specified mirrors and selects the first one that appears usable. This new behavior is only activated when the `primary` section is wrapped in the `mirror-selection` section.
- The `fallback` key controls what Subiquity does when no primary mirror is usable.
- The `geoip` key controls whether to perform IP-based geolocation to determine the correct country mirror.

All other sections behave as defined in curtin. See the curtin documentation and its example apt configurations for usage examples of these sections, such as how to add a PPA using the `sources` section.

The default configuration in Subiquity is equivalent to:

```
autoinstall:
  apt:
    preserve_sources_list: false
    mirror-selection:
      primary:
        - country-mirror
        - uri: "http://archive.ubuntu.com/ubuntu"
          arches: [i386, amd64]
        - uri: "http://ports.ubuntu.com/ubuntu-ports"
          arches: [s390x, arm64, armhf, powerpc, ppc64el, riscv64]
    fallback: abort
    geoip: true
```

mirror-selection

If the `primary` section is contained within the `mirror-selection` section, the automatic mirror selection is enabled. This is the default in new installations.

primary (when placed inside the `mirror-selection` section)

- **type:** custom, see below

In the new format, the `primary` section expects a list of mirrors, which can be expressed in two different ways:

- The special `country-mirror` value
- A mapping with the following keys:
  - `uri` (Required): The URI of the mirror to use, e.g., `http://fr.archive.ubuntu.com/ubuntu`.
  - `arches` (Optional): A list of architectures supported by the mirror. By default, this list contains the current CPU architecture.

The URI for the archive mirror does not have to be a country mirror, although it may be the most convenient, and can take the URL of any valid Ubuntu mirror. A list of all registered archive mirrors can be found on Launchpad.

Examples:

```
# Use the first custom mirror that works. Do not restrict to specific architec
autoinstall:
  apt:
    mirror-selection:
      primary:
        - uri: "http://mirror1.internal/ubuntu"
        - uri: "http://mirror2.internal/ubuntu"

# Use one mirror for amd64 and another for i386.
autoinstall:
  apt:
    mirror-selection:
      primary:
        - uri: "http://jp.archive.ubuntu.com/ubuntu"
          arches: [amd64]
        - uri: "http://tw.archive.ubuntu.com/ubuntu"
          arches: [i386]
```

fallback

- **type:** string (enumeration)
- **default:** `offline-install`

Controls what Subiquity does when no primary mirror is usable. Supported values are:

- `abort`: abort the installation
- `offline-install`: revert to an offline installation
- `continue-anyway`: attempt to install the system anyway (not recommended; the installation fails)

Examples:

```
# Only install from the primary archive and abort the installation if mirror v
autoinstall:
  apt:
    mirror-selection:
      primary:
        - uri: "http://archive.ubuntu.com/ubuntu"
    fallback: abort

# Only install from the German country mirror and continue with an offline ins
autoinstall:
  apt:
    mirror-selection:
      primary:
        - uri: "http://de.archive.ubuntu.com/ubuntu"
    fallback: offline-install
```

geoip

- **type:** boolean
- **default:** `true`

If `geoip` is set to `true` and one of the candidate primary mirrors has the special value `country-mirror`, a request is made to `https://geoip.ubuntu.com/lookup`. Subiquity then sets the mirror URI to `http://CC.archive.ubuntu.com/ubuntu` where `CC` is the country code returned by the lookup. If this section is not interactive, the request expires after 10 seconds.

If the legacy behavior (i.e., without mirror-selection) is in use, the geolocation request is made if the mirror to be used is the default, and its URI is replaced by the proper country mirror URI.

Examples:

```
# Use the automatically determined country mirror first, followed by an explic
autoinstall:
  apt:
    mirror-selection:
      primary:
        - country-mirror
        - uri: http://dk.archive.ubuntu.com/ubuntu
    geoip: true

# Disable automatic country mirror detection (i.e. only use http://archive.ubu
autoinstall:
  apt:
    geoip: false
```

storage

- **type:** mapping, see below
- **default:** use the `lvm` layout on single-disk systems; there is no default for multiple-disk systems
- **can be interactive:** true

Storage configuration is a complex topic, and the description of the desired configuration in the autoinstall file can also be complex. The installer supports “layouts”; simple ways of expressing common configurations.

Supported layouts

The three supported layouts at the time of writing are `lvm`, `direct` and `zfs`.

```
autoinstall:
  storage:
    layout:
      name: lvm
  storage:
    layout:
      name: direct
  storage:
    layout:
      name: zfs
```

By default, these layouts install to the largest disk in a system, but you can supply a match spec (see below) to indicate which disk to use:

```
autoinstall:
  storage:
    layout:
      name: lvm
      match:
        serial: CT*
  storage:
    layout:
      name: direct
      match:
        ssd: true
```

> **Note**
>
> Match spec – using `match: {}` matches an arbitrary disk.

By default (except on s390x), the matching disk will be partitioned using a GUID Partition Table (GPT). But you can specifically request a MSDOS (aka. MBR) partition table:

```
autoinstall:
  storage:
    layout:
      name: direct
      ptable: msdos
```

When using the `lvm` layout, LUKS encryption can be enabled by supplying a password.

```
autoinstall:
  storage:
    layout:
      name: lvm
      password: LUKS_PASSPHRASE
```

The default is to use the `lvm` layout.

-backed encryption can be enabled by using the `hybrid` layou

set to yes.

```
autoinstall:
  storage:
    layout:
      name: hybrid
      encrypted: yes
```

Sizing-policy

The `lvm` layout, by default, attempts to leave room for snapshots and further expansion. A sizing-policy key may be supplied to control this behavior.

- **type:** string (enumeration)
- **default:** scaled

Supported values are:

- `scaled`: Adjust space allocated to the root logical volume (LV) based on space available to the volume group (VG).
- `all`: Allocate all remaining VG space to the root LV.

The scaling system uses the following rules:

- Less than 10 GiB: use all remaining space for the root file system
- Between 10–20 GiB: 10 GiB root file system
- Between 20–200 GiB: use half of the remaining space for the root file system
- Greater than 200 GiB: 100 GiB root file system

Example with no size scaling and a passphrase:

```
autoinstall:
  storage:
    layout:
      name: lvm
      sizing-policy: all
      password: LUKS_PASSPHRASE
```

Reset Partition

`reset-partition` is used for creating a Reset Partition, which is a FAT32 file system containing

: of the installer image, so that the user can start the installer without using the installation media. This option is useful for OEM system provisioning.

By default, the size of a Reset Partition is roughly 1.1x the used file system size of the installation media.

An example to enable Reset Partition:

```
autoinstall:
  storage:
    layout:
      name: direct
      reset-partition: true
```

The size of the reset partition can also be fixed to a specified size. This is an example to fix Reset Partition to 12 GiB:

```
autoinstall:
  storage:
    layout:
      name: direct
      reset-partition: 12G
```

The installer can also install Reset Partition without installing the system. To do this, set `reset-partition-only` to `true`:

```
autoinstall:
  storage:
    layout:
      name: direct
      reset-partition: true
      reset-partition-only: true
```

Action-based configuration

For full flexibility, the installer allows storage configuration to be done using a syntax that is a superset of that supported by curtin, as described in the Storage section of the curtin documentation.

If the `layout` feature is used to configure the disks, the `config` section is not used.

can be added under the `config` key, and the grub and swap curtin configuration ncluded here.

An example storage section:

```
autoinstall:
  storage:
    swap:
      size: 0
    config:
      - type: disk
        id: disk0
        serial: ADATA_SX8200PNP_XXXXXXXXXXX
      - type: partition
        ...
```

The extensions to the curtin syntax allow for disk selection and partition or logical-volume sizing.

Disk selection extensions

Curtin supported identifying disks by serial numbers (e.g. `Crucial_CT512MX100SSD1_14250C57FECE`) or by path (e.g. `/dev/sdc`), and the server installer supports this, too. The installer additionally supports a "match spec" on a disk action, which provides for more flexible matching.

The actions in the storage configuration are processed in the order they are in the autoinstall file. Any disk action is assigned a matching disk – chosen arbitrarily from the set of unassigned disks if there is more than one, and causing the installation to fail if there is no unassigned matching disk.

A match spec supports the following keys:

- `model: value`: matches a disk where `ID_MODEL=value` in udev, supporting globbing
- `vendor: value`: matches a disk where `ID_VENDOR=value` in udev, supporting globbing
- `path: value`: matches a disk based on path (e.g. `/dev/sdc`), supporting globbing (the globbing support distinguishes this from specifying `path: value` directly in the disk action)
- `id_path: value`: matches a disk where `ID_PATH=value` in udev, supporting globbing
- `devpath: value`: matches a disk where `DEVPATH=value` in udev, supporting globbing
- `serial: value`: matches a disk where `ID_SERIAL=value` in udev, supporting globbing (the globbing support distinguishes this from specifying `serial: value` directly in the disk action)
- `ssd: true|false`: matches a disk that is or is not an SSD (as opposed to a rotating drive)
- `size: largest|smallest`: take the largest or smallest disk rather than an arbitrary one if there are multiple matches (support for `smallest` added in version 20.06.1)

ey is `install-media: true`, which takes the disk the install
e selectors never return this disk). If installing to the installat

careful to not overwrite the installer itself.

For example, to match an arbitrary disk:

```
- type: disk
  id: disk0
```

To match the largest SSD:

```
- type: disk
  id: big-fast-disk
  match:
    ssd: true
    size: largest
```

To match a Seagate drive:

```
- type: disk
  id: data-disk
  match:
    model: Seagate
```

As of Subiquity 24.08.1, match specs may optionally be specified in an ordered list, and will use the first match spec that matches one or more unused disks:

```
# attempt first to match by serial, then by path
- type: disk
  id: data-disk
  match:
    - serial: Foodisk_1TB_ABC123_1
    - path: /dev/nvme0n1
```

Partition/logical volume extensions

The size of a partition or logical volume in curtin is specified as a number of bytes. The autoinstall configuration is more flexible:

- You can specify the size using the `1G`, `512M` syntax supported in the installer UI.
- You can specify the size as a percentage of the containing disk (or RAID), e.g. `50%`.
- You can specify the size of a partition specified for a particular device, you can specify the size as `-1` to indicate that the partition should fill the remaining space.

```
- type: partition
  id: boot-partition
  device: root-disk
  size: 10%
- type: partition
  id: root-partition
  size: 20G
- type: partition
  id: data-partition
  device: root-disk
  size: -1
```

identity

- **type:** mapping, see below
- **default:** no default
- **can be interactive:** true

Configure the initial user for the system. This is the only configuration key that must be present (unless the user-data section is present, in which case it is optional).

A mapping that can contain keys, all of which but groups take string values:

realname

The real name for the user. This field is optional.

username

The user name to create.

hostname

The hostname for the system.

password

The password for the new user, encrypted. This is required for use with `sudo`, even if SSH access is configured.

The encrypted password string must conform to what the `passwd` command requires. See the passwd(1) manual page for details. Quote the password hash to ensure correct treatment of any

Several tools can generate the encrypted password, such as `mkpasswd` from the `whois` package, or `openssl passwd`.

groups

- **type:** mapping or list of strings (see below)
- **default:** hard-coded list of groups (e.g., *sudo*, *admin*)

Configures which groups the newly created user should belong to. The hard-coded groups (i.e., *sudo*, *admin*) can either be included or excluded.

The mapping contains the *override* and *append* keys, which are mutually exclusive. The groups directive also accepts a list of strings, providing syntactic sugar for *{override: …}*.

- *override*

Specifies the list of groups that the user should belong to, ignoring groups hard-coded in Subiquity.

- *append*

Specifies the list of groups that the user should belong to, in addition to the defaults.

Example:

```
autoinstall:
  identity:
    realname: 'Ubuntu User'
    username: ubuntu
    password: '$6$wdAcoXrU039hKYPd$508Qvbe7ObUnxoj15DRCkzC3qO7edjH0VV7BPNRDYK4
    hostname: ubuntu

autoinstall:
  identity:
    username: ubuntu
    password: '$6$wdAcoXrU039hKYPd$508Qvbe7ObUnxoj15DRCkzC3qO7edjH0VV7BPNRDYK4
    hostname: ubuntu
    # Specifies the list of groups the user should belong to.
    groups: [adm, sudo, lpadmin]
    # Alternative syntax:
    # groups:
    #   override: [adm, sudo, lpadmin]

autoinstall:
  identity:
    username: ubuntu
    password: '$6$wdAcoXrU039hKYPd$508Qvbe7ObUnxoj15DRCkzC3qO7edjH0VV7BPNRDYK4
    hostname: ubuntu
    # Ensure the user is part of supplementary groups.
    groups:
      append: [adm, sudo, lpadmin]
```

Note that in any case, one more group will be created for the user, with the same name as their username (see *useradd(8)*). This behavior cannot currently be disabled.

active-directory

- **type:** mapping, see below
- **default:** no default
- **can be interactive:** true

ired to join the target system in an Active Directory domain.

A mapping that can contain keys, all of which take string values:

admin-name

A domain account name with the privilege to perform the join operation. The account password is requested during run time.

domain-name

The Active Directory domain to join.

Example:

```
autoinstall:
  active-directory:
    # Join the Active Directory domain as user "$ubuntu"
    admin-name: $ubuntu
    domain-name: ad.ubuntu.com
```

ubuntu-pro

- **type:** mapping, see below
- **default:** see below
- **can be interactive:** true

token

- **type:** string
- **default:** no token

A contract token to attach to an existing Ubuntu Pro subscription.

Example:

```
autoinstall:
  ubuntu-pro:
    # Enable Ubuntu-Pro using a contract token
    # Note that the example below is an invalid contract token.
    token: C1NWcZTHLteJXGVMM6YhvHDpGrhyy7
```

ssh

- **type:** mapping, see below
- **default:** see below
- **can be interactive:** true

Configure SSH for the installed system. A mapping that can contain the following keys:

install-server

- **type:** boolean
- **default:** `false`

Whether to install the OpenSSH server in the target system. Note that Desktop installation ISOs do not include `openssh-server`, so installations of Desktop require Ubuntu archive access for `install-server` to be successful.

authorized-keys

- **type:** list of strings
- **default:** `[]`

A list of SSH public keys to install in the initial user account.

allow-pw

- **type:** boolean
- **default:** `true` if `authorized_keys` is empty, `false` otherwise

`ssh` examples:

```
autoinstall:
  # default behavior
  ssh:
    install-server: false
    authorized-keys: []
    allow-pw: true

autoinstall:
  # recommended configuration when openssh-server is desired
  ssh:
    install-server: true
    authorized-keys:
      # replace with the contents of the public key(s) as generated by
      # ssh-keygen or similar tools
      - ssh-ed25519 AAAAC3NzaC..6O8tvZobj user@host
    allow-pw: false

autoinstall:
  # configuration for password access
  ssh:
    install-server: true
    allow-pw: true
```

codecs

- **type:** mapping, see below
- **default:** see below
- **can be interactive:** no

Configure whether common restricted packages (including codecs) from the multiverse repository are to be installed.

install

- **type:** boolean
- **default:** `false`

Skip to content ` the `ubuntu-restricted-addons` package.

latest

Examples:

```
autoinstall:
  # default behavior
  codecs:
    install: false

autoinstall:
  # install codecs, which currently means installing the
  # ubuntu-restricted-addons package
  codecs:
    install: true
```

drivers

- **type:** mapping, see below
- **default:** see below
- **can be interactive:** true

install

- **type:** boolean
- **default:** `false`

Whether to install the available third-party drivers.

Examples:

```
autoinstall:
  # default behavior
  drivers:
    install: false

autoinstall:
  # install drivers as suggested by `ubuntu-drivers`.
  drivers:
    install: true
```

oem

- **type:** mapping, see below
- **default:** see below
- **can be interactive:** no

install

- **type:** boolean or string (special value `auto`)
- **default:** `auto`

Whether to install the available OEM meta-packages. The special value `auto` – which is the default – enables the installation on Ubuntu Desktop but not on Ubuntu Server. This option has no effect on core boot classic.

As installing an OEM meta-package can result in installing a certain kernel, specifying both a kernel with kernel and also specifying `oem.install: true` may lead to an install failure due to conflicting kernel requirements. When using oem.install, it is recommended to not specify a kernel.

Examples:

```
autoinstall:
  # default behavior
  oem:
    install: auto

autoinstall:
  # Install OEM meta-packages as suggested by ubuntu-drivers.
  # On some hardware, this changes what kernel is installed.
  oem:
    install: true

autoinstall:
  # Disable OEM meta-package automatic installation, even if suggested to do
  # so by ubuntu-drivers
  oem:
    install: false
```

snaps

- **type:** list
- **default:** install no extra snaps
- **can be interactive:** true

A list of snaps to install. Each snap is represented as a mapping with a required `name` and an optional `channel` (default is `stable`) and classic (default is `false`) keys. For example:

```
autoinstall:
  snaps:
    - name: etcd
      channel: edge
      classic: false
```

debconf-selections

- **type:** string
- **default:** no configuration
- **can be interactive:** no

The installer updates the target with debconf `set-selection` values. Users need to be familiar with the options of the `debconf` package.

Example:

```
autoinstall:
  # 1. Make sure history files are purged if one runs `apt purge etckeeper`
  # 2. Disable SSH password authentication for root (which is actually the
  # default behavior)
  # Watch out, permit-root-login does basically the opposite of
  # PermitRootLogin (see man 8 sshd_config):
  #  * openssh-server/permit-root-login boolean false  <=>  PermitRootLogin ye
  #  * openssh-server/permit-root-login boolean true   <=>  # PermitRootLogin
  # See https://salsa.debian.org/ssh-team/openssh/-/blob/78b01f70b043846640887
  # and LP: #2128863
  # 3. Start the ufw firewall automatically.

  debconf-selections: |
    etckeeper etckeeper/purge boolean true
    openssh-server openssh-server/permit-root-login boolean true
    ufw ufw/enable boolean true
```

packages

- **type:** list
- **default:** no packages
- **can be interactive:** no

A list of packages to install into the target system. Specifically, a list of strings to pass to the `apt-get install` command. Therefore, this includes things such as task selection (`dns-server^`) and installing particular versions of a package (`my-package=1-1`).

Example:

```
autoinstall:
  packages:
    # Install ipython3 and git, and ensure they are marked as manually
    # installed.
    - ipython3
    - git
```

kernel

- **type:** mapping (mutually exclusive), see below
- **default:** default kernel
- **can be interactive:** no

Which kernel gets installed. Either the name of the package or the name of the flavor must be specified.

The exact default kernel is ISO build specific, but generally the `generic` flavor is installed for Server and the `hwe` flavor is installed for Desktop.

package

type: string

The name of the package, e.g., `linux-image-5.13.0-40-generic`.

flavor

- **type:** string

The `flavor` of the kernel, e.g., `generic` or `hwe`.

Example:

```
autoinstall:
   # Install a specific kernel package.
   kernel:
     package: linux-image-5.13.0-40-generic

autoinstall:
   # Install a particular kernel flavor.
   kernel:
     flavor: hwe
```

kernel-crash-dumps

- **type:** mapping, see below
- **default:** see below

:**ive:** no

Toggle kernel crash dumps enablement.

In 24.10 and later, the default configuration will result in dynamic enablement of kernel crash dumps on the installed system using the `kdump-tools` package. On amd64, arm64, and s390x systems, if the system is detected to meet the minimum requirements for kernel crash dumps then they will be enabled. Otherwise, they will be disabled. More details about the minimum system requirements can be found in the Ubuntu Server documentation.

In pre-24.10, the default configuration will result in kernel crash dumps being disabled.

Default configuration:

```
autoinstall:
   # In 24.10 and later, allow kernel crash dumps to be enabled dynamically.
   # In pre-24.10, kernel crash dumps will be disabled.
   kernel-crash-dumps:
     enabled: null
```

enabled

- **type:** boolean or null
- **default:** `null`

Specify a boolean value to enable or disable kernel crash dumps. Set to `null` (default) to allow dynamic enablement.

If kernel crash dumps are to be disabled, whether determined dynamically or manually requested, the `kdump-tools` package will not be uninstalled but will be configured to ensure it is inactive in the target system.

Examples:

```
autoinstall:
   # Enable kernel crash dumps.
   kernel-crash-dumps:
     enabled: true

autoinstall:
   # Disable kernel crash dumps.
   kernel-crash-dumps:
     enabled: false
```

timezone

- **type:** string
- **default:** no timezone
- **can be interactive:** no

The timezone to configure on the system.

timezone examples:

```
autoinstall:
   # Default behavior
   timezone: "Etc/UTC"

autoinstall:
   # Configure explicitly
   timezone: "Europe/London"
```

updates

- **type:** string (enumeration)
- **default:** `security`
- **can be interactive:** no

The type of updates that will be downloaded and installed after the system installation, and before rebooting into the target system. Supported values are:

- `security`: download and install updates from the `-security` pocket.
- `all`: also download and install updates from the `-updates` pocket.

Examples:

```
autoinstall:
  # default behavior.  Updates from the security pocket are installed.
  updates: security

autoinstall:
  # Updates from both the security and updates pockets are installed.
  updates: all
```

shutdown

- **type:** string (enumeration)
- **default:** `reboot`
- **can be interactive:** no

Request the system to power off or reboot automatically after the installation has finished. Supported values are:

- `reboot`
- `poweroff`

shutdown examples:

```
autoinstall:
  # default behavior
  shutdown: reboot

autoinstall:
  # shutdown instead of reboot
  shutdown: poweroff
```

late-commands

- **type:** command list
- **default:** no commands
- **can be interactive:** no

Shell commands to run after the installation has completed successfully and any updates and packages installed, just before the system reboots. The commands are run in the installer environment with the installed system mounted at `/target`. You can run `curtin in-target -- $shell_command` to run in the target system (similar to how plain `in-target` can be used in `d-i preseed/late_command`).

Example late commands:

```
autoinstall:
  # Pause the install just before finishing to allow manual inspection/modific
  # Unpause by creating the "/run/finish-late" file.
  late-commands:
    - while [ ! -f /run/finish-late ]; do sleep 1; done

autoinstall:
  # Install additional packages on the target system and run some custom scrip
  late-commands:
    - curtin in-target -- apt-get update
    - curtin in-target -- apt-get install -y curl vim
    - curtin in-target -- curl -o /tmp/my-script.sh $script_url
    - curtin in-target -- /bin/sh /tmp/my-script.sh
```

error-commands

- **type:** command list
- **default:** no commands
- **can be interactive:** no

Shell commands to run after the installation has failed. They are run in the installer environment, and the target system (or as much of it as the installer managed to configure) is mounted at `/target`. Logs will be available in `/var/log/installer` in the live session.

```
autoinstall:
  # Collect all of the logs in /var/log/installer
  # Collect the live system journal too
  error-commands:
    - tar -czf /installer-logs.tar.gz /var/log/installer/
    - journalctl -b > /installer-journal.log
```

reporting

- **type:** mapping
- **default:** `type: print` (which causes output on `tty1` and any configured serial consoles)
- **can be interactive:** no

The installer supports reporting progress to a variety of destinations. Note that this section is ignored if there are any interactive sections; it only applies to fully automated installations.

The configuration is similar to that used by curtin. See the Reporting section of the curtin documentation.

Each key in the `reporting` mapping in the configuration defines a destination where the `type` sub-key is one of:

- `print`: print progress information on `tty1` and any configured serial console. There is no other configuration.
- `rsyslog`: report progress via rsyslog. The `destination` key specifies where to send output. (The rsyslog reporter does not yet exist.)
- `webhook`: report progress by sending JSON reports to a URL using POST requests. Accepts the same configuration as curtin.
- `none`: do not report progress. Only useful to inhibit the default output.

Reporting examples:

The default configuration is:

```
autoinstall:
  reporting:
    builtin:
      type: print
```

Report to rsyslog:

```
autoinstall:
  reporting:
    central:
      type: rsyslog
      destination: "@192.168.0.1"
```

Suppress the default output:

```
autoinstall:
  reporting:
    builtin:
      type: none
```

```
autoinstall:
  reporting:
    hook:
      type: webhook
      endpoint: http://example.com/endpoint/path
      consumer_key: "ck_value"
      consumer_secret: "cs_value"
      token_key: "tk_value"
      token_secret: "tk_secret"
      level: INFO
```

user-data

- **type:** mapping
- **default:** `{}`
- **can be interactive:** no

Provide cloud-init user data, which will be merged with the user data that the installer may produce. If you supply this, you don't need to supply an identity section (in that case, ensure you can log in to the installed system). For more details on cloud-init user-data, see All cloud config examples.

The following example uses user-data and is roughly equivalent to what the identity section can be used for. The main difference is the timing of user creation. While users specified using the identity section are created during the installation, users created using cloud-init user-data will be created on first boot.

```
autoinstall:
  user-data:
     users:
       - name: ubuntu
         gecos: 'Ubuntu User'
         passwd: '$6$wdAcoXrU039hKYPd$508Qvbe7ObUnxoj15DRCkzC3qO7edjH0VV7BPNRD
         groups: adm, cdrom, dip, lxd, plugdev, sudo
         shell: /bin/bash
         lock_passwd: False
```

zdevs

- **type:** list of devices
- **default:** `[]`
- **can be interactive:** yes

On IBM Z, configure the state (enabled or disabled) of supported devices.

Each element (i.e., device) of the list is a mapping with the following keys:

id

type: string

Identifies the device to operate on.

enabled

type: boolean

Controls whether the device should be enabled or disabled.

```
autoinstall:
  zdevs:
    - id: 0.0.1507
      enabled: true
    - id: 0.0.1508
      enabled: true
    - id: 0.0.1509
      enabled: false
```