



## ***Rocky Enterprise Linux 9.2 Manual Pages on command 'machinectl.1'***

**C:\>man machinectl.1**

MACHINECTL(1)                      machinectl                      MACHINECTL(1)

### NAME

machinectl - Control the systemd machine manager

### SYNOPSIS

machinectl [OPTIONS...] {COMMAND} [NAME...]

### DESCRIPTION

machinectl may be used to introspect and control the state of the systemd(1)

virtual machine and container registration manager systemd-machined.service(8).

machinectl may be used to execute operations on machines and images. Machines in this sense are considered running instances of:

? Virtual Machines (VMs) that virtualize hardware to run full operating system (OS) instances (including their kernels) in a virtualized environment on top of the host OS.

? Containers that share the hardware and OS kernel with the host OS, in order to run OS userspace instances on top the host OS.

? The host system itself.

Machines are identified by names that follow the same rules as UNIX and DNS host names. For details, see below.

Machines are instantiated from disk or file system images that frequently ? but not necessarily ? carry the same name as machines running from them. Images in this sense may be:

? Directory trees containing an OS, including the top-level directories /usr,

/etc, and so on.

- ? btrfs subvolumes containing OS trees, similar to regular directory trees.
- ? Binary "raw" disk image files containing MBR or GPT partition tables and Linux file systems.
- ? Similarly, block devices containing MBR or GPT partition tables and file systems.
- ? The file system tree of the host OS itself.

## COMMANDS

The following commands are understood:

### Machine Commands

`list`

List currently running (online) virtual machines and containers. To enumerate machine images that can be started, use `list-images` (see below). Note that this command hides the special ".host" machine by default. Use the `--all` switch to show it.

`status NAME...`

Show runtime status information about one or more virtual machines and containers, followed by the most recent log data from the journal. This function is intended to generate human-readable output. If you are looking for computer-parsable output, use `show` instead. Note that the log data shown is reported by the virtual machine or container manager, and frequently contains console output of the machine, but not necessarily journal contents of the machine itself.

`show [NAME...]`

Show properties of one or more registered virtual machines or containers or the manager itself. If no argument is specified, properties of the manager will be shown. If a NAME is specified, properties of this virtual machine or container are shown. By default, empty properties are suppressed. Use `--all` to show those too. To select specific properties to show, use `--property=`. This command is intended to be used whenever computer-parsable output is required, and does not print the control group tree or journal entries. Use `status` if you are looking for formatted human-readable output.

`start NAME...`

Start a container as a system service, using `systemd-nspawn(1)`. This starts `systemd-nspawn@.service`, instantiated for the specified machine name, similar to the effect of `systemctl start` on the service name. `systemd-nspawn` looks for a container image by the specified name in `/var/lib/machines/` (and other search paths, see below) and runs it. Use `list-images` (see below) for listing available container images to start.

Note that `systemd-machined.service(8)` also interfaces with a variety of other container and VM managers, `systemd-nspawn` is just one implementation of it. Most of the commands available in `machinectl` may be used on containers or VMs controlled by other managers, not just `systemd-nspawn`. Starting VMs and container images on those managers requires manager-specific tools.

To interactively start a container on the command line with full access to the container's console, please invoke `systemd-nspawn` directly. To stop a running container use `machinectl poweroff`.

#### `login [NAME]`

Open an interactive terminal login session in a container or on the local host. If an argument is supplied, it refers to the container machine to connect to. If none is specified, or the container name is specified as the empty string, or the special machine name `".host"` (see below) is specified, the connection is made to the local host instead. This will create a TTY connection to a specific container or the local host and asks for the execution of a `getty` on it. Note that this is only supported for containers running `systemd(1)` as init system. This command will open a full login prompt on the container or the local host, which then asks for username and password. Use `shell` (see below) or `systemd-run(1)` with the `--machine=` switch to directly invoke a single command, either interactively or in the background.

#### `shell [[NAME@]NAME [PATH [ARGUMENTS...]]]`

Open an interactive shell session in a container or on the local host. The first argument refers to the container machine to connect to. If none is specified, or the machine name is specified as the empty string, or the special machine name `".host"` (see below) is specified, the connection is made to the local host instead. This works similar to `login` but immediately invokes a user process. This command runs the specified executable with the specified

arguments, or the default shell for the user if none is specified, or `/bin/sh` if no default shell is found. By default, `--uid=`, or by prefixing the machine name with a username and an "@" character, a different user may be selected. Use `--setenv=` to set environment variables for the executed process.

Note that `machinectl shell` does not propagate the exit code/status of the invoked shell process. Use `systemd-run` instead if that information is required (see below).

When using the shell command without arguments, (thus invoking the executed shell or command on the local host), it is in many ways similar to a `su(1)` session, but, unlike `su`, completely isolates the new session from the originating session, so that it shares no process or session properties, and is in a clean and well-defined state. It will be tracked in a new `utmp`, `login`, `audit`, `security` and `keyring` session, and will not inherit any environment variables or resource limits, among other properties.

Note that `systemd-run(1)` with its `--machine=` switch may be used in place of the `machinectl shell` command, and allows non-interactive operation, more detailed and low-level configuration of the invoked unit, as well as access to runtime and exit code/status information of the invoked shell process. In particular, use `systemd-run's --wait` switch to propagate exit status information of the invoked process. Use `systemd-run's --pty` switch for acquiring an interactive shell, similar to `machinectl shell`. In general, `systemd-run` is preferable for scripting purposes. However, note that `systemd-run` might require higher privileges than `machinectl shell`.

`enable NAME...`, `disable NAME...`

Enable or disable a container as a system service to start at system boot, using `systemd-nspawn(1)`. This enables or disables `systemd-nspawn@.service`, instantiated for the specified machine name, similar to the effect of `systemctl enable` or `systemctl disable` on the service name.

`poweroff NAME...`

Power off one or more containers. This will trigger a reboot by sending `SIGRTMIN+4` to the container's `init` process, which causes `systemd-compatible` `init` systems to shut down cleanly. Use `stop` as alias for `poweroff`. This operation does not work on containers that do not run a `systemd(1)-compatible`

init system, such as sysvinit. Use terminate (see below) to immediately terminate a container or VM, without cleanly shutting it down.

reboot NAME...

Reboot one or more containers. This will trigger a reboot by sending SIGINT to the container's init process, which is roughly equivalent to pressing Ctrl+Alt+Del on a non-containerized system, and is compatible with containers running any system manager.

terminate NAME...

Immediately terminates a virtual machine or container, without cleanly shutting it down. This kills all processes of the virtual machine or container and deallocates all resources attached to that instance. Use poweroff to issue a clean shutdown request.

kill NAME...

Send a signal to one or more processes of the virtual machine or container. This means processes as seen by the host, not the processes inside the virtual machine or container. Use --kill-who= to select which process to kill. Use --signal= to select the signal to send.

bind NAME PATH [PATH]

Bind mounts a file or directory from the host into the specified container. The first path argument is the source file or directory on the host, the second path argument is the destination file or directory in the container. When the latter is omitted, the destination path in the container is the same as the source path on the host. When combined with the --read-only switch, a read-only bind mount is created. When combined with the --mkdir switch, the destination path is first created before the mount is applied. Note that this option is currently only supported for systemd-nspawn(1) containers, and only if user namespacing (--private-users) is not used. This command supports bind mounting directories, regular files, device nodes, AF\_UNIX socket nodes, as well as FIFOs.

copy-to NAME PATH [PATH]

Copies files or directories from the host system into a running container. Takes a container name, followed by the source path on the host and the destination path in the container. If the destination path is omitted, the same

as the source path is used.

If host and container share the same user and group namespace, file ownership by numeric user ID and group ID is preserved for the copy, otherwise all files and directories in the copy will be owned by the root user and group (UID/GID 0).

`copy-from NAME PATH [PATH]`

Copies files or directories from a container into the host system. Takes a container name, followed by the source path in the container the destination path on the host. If the destination path is omitted, the same as the source path is used.

If host and container share the same user and group namespace, file ownership by numeric user ID and group ID is preserved for the copy, otherwise all files and directories in the copy will be owned by the root user and group (UID/GID 0).

## Image Commands

`list-images`

Show a list of locally installed container and VM images. This enumerates all raw disk images and container directories and subvolumes in `/var/lib/machines/` (and other search paths, see below). Use `start` (see above) to run a container off one of the listed images. Note that, by default, containers whose name begins with a dot (".") are not shown. To show these too, specify `--all`. Note that a special image `".host"` always implicitly exists and refers to the image the host itself is booted from.

`image-status [NAME...]`

Show terse status information about one or more container or VM images. This function is intended to generate human-readable output. Use `show-image` (see below) to generate computer-parsable output instead.

`show-image [NAME...]`

Show properties of one or more registered virtual machine or container images, or the manager itself. If no argument is specified, properties of the manager will be shown. If a `NAME` is specified, properties of this virtual machine or container image are shown. By default, empty properties are suppressed. Use `--all` to show those too. To select specific properties to show, use

`--property=`. This command is intended to be used whenever computer-parsable output is required. Use `image-status` if you are looking for formatted human-readable output.

`clone NAME NAME`

Clones a container or VM image. The arguments specify the name of the image to clone and the name of the newly cloned image. Note that plain directory container images are cloned into btrfs subvolume images with this command, if the underlying file system supports this. Note that cloning a container or VM image is optimized for file systems that support copy-on-write, and might not be efficient on others, due to file system limitations.

Note that this command leaves host name, machine ID and all other settings that could identify the instance unmodified. The original image and the cloned copy will hence share these credentials, and it might be necessary to manually change them in the copy.

If combined with the `--read-only` switch a read-only cloned image is created.

`rename NAME NAME`

Renames a container or VM image. The arguments specify the name of the image to rename and the new name of the image.

`read-only NAME [BOOL]`

Marks or (unmarks) a container or VM image read-only. Takes a VM or container image name, followed by a boolean as arguments. If the boolean is omitted, positive is implied, i.e. the image is marked read-only.

`remove NAME...`

Removes one or more container or VM images. The special image `".host"`, which refers to the host's own directory tree, may not be removed.

`set-limit [NAME] BYTES`

Sets the maximum size in bytes that a specific container or VM image, or all images, may grow up to on disk (disk quota). Takes either one or two parameters. The first, optional parameter refers to a container or VM image name. If specified, the size limit of the specified image is changed. If omitted, the overall size limit of the sum of all images stored locally is changed. The final argument specifies the size limit in bytes, possibly suffixed by the usual K, M, G, T units. If the size limit shall be disabled,

specify "-" as size.

Note that per-container size limits are only supported on btrfs file systems.

## clean

Remove hidden VM or container images (or all). This command removes all hidden machine images from `/var/lib/machines`, i.e. those whose name begins with a dot.

Use `machinectl list-images --all` to see a list of all machine images, including the hidden ones.

When combined with the `--all` switch removes all images, not just hidden ones.

This command effectively empties `/var/lib/machines`.

Note that commands such as `machinectl pull-tar` or `machinectl pull-raw` usually create hidden, read-only, unmodified machine images from the downloaded image first, before cloning a writable working copy of it, in order to avoid duplicate downloads in case of images that are reused multiple times. Use `machinectl clean` to remove old, hidden images created this way.

## Image Transfer Commands

### pull-tar URL [NAME]

Downloads a `.tar` container image from the specified URL, and makes it available under the specified local machine name. The URL must be of type `"http://"` or `"https://"`, and must refer to a `.tar`, `.tar.gz`, `.tar.xz` or `.tar.bz2` archive file. If the local machine name is omitted, it is automatically derived from the last component of the URL, with its suffix removed.

The image is verified before it is made available, unless `--verify=no` is specified. Verification is done either via an inline signed file with the name of the image and the suffix `.sha256` or via separate `SHA256SUMS` and `SHA256SUMS.gpg` files. The signature files need to be made available on the same web server, under the same URL as the `.tar` file. With `--verify=checksum`, only the SHA256 checksum for the file is verified, based on the `.sha256` suffixed file or the `SHA256SUMS` file. With `--verify=signature`, the sha checksum file is first verified with the inline signature in the `.sha256` file or the detached GPG signature file `SHA256SUMS.gpg`. The public key for this verification step needs to be available in `/usr/lib/systemd/import-pubring.gpg` or `/etc/systemd/import-pubring.gpg`.

The container image will be downloaded and stored in a read-only subvolume in

`/var/lib/machines/` that is named after the specified URL and its HTTP etag. A writable snapshot is then taken from this subvolume, and named after the specified local name. This behavior ensures that creating multiple container instances of the same URL is efficient, as multiple downloads are not necessary. In order to create only the read-only image, and avoid creating its writable snapshot, specify "-" as local machine name.

Note that the read-only subvolume is prefixed with `.tar-`, and is thus not shown by `list-images`, unless `--all` is passed.

Note that pressing C-c during execution of this command will not abort the download. Use `cancel-transfer`, described below.

#### `pull-raw URL [NAME]`

Downloads a `.raw` container or VM disk image from the specified URL, and makes it available under the specified local machine name. The URL must be of type `"http://"` or `"https://"`. The container image must either be a `.qcow2` or raw disk image, optionally compressed as `.gz`, `.xz`, or `.bz2`. If the local machine name is omitted, it is automatically derived from the last component of the URL, with its suffix removed.

Image verification is identical for raw and tar images (see above).

If the downloaded image is in `.qcow2` format it is converted into a raw image file before it is made available.

Downloaded images of this type will be placed as read-only `.raw` file in `/var/lib/machines/`. A local, writable (reflinked) copy is then made under the specified local machine name. To omit creation of the local, writable copy pass "-" as local machine name.

Similar to the behavior of `pull-tar`, the read-only image is prefixed with `.raw-`, and thus not shown by `list-images`, unless `--all` is passed.

Note that pressing C-c during execution of this command will not abort the download. Use `cancel-transfer`, described below.

#### `import-tar FILE [NAME]`, `import-raw FILE [NAME]`

Imports a TAR or RAW container or VM image, and places it under the specified name in `/var/lib/machines/`. When `import-tar` is used, the file specified as the first argument should be a tar archive, possibly compressed with `xz`, `gzip` or `bzip2`. It will then be unpacked into its own subvolume in `/var/lib/machines`.

When `import-raw` is used, the file should be a `qcow2` or raw disk image, possibly compressed with `xz`, `gzip` or `bzip2`. If the second argument (the resulting image name) is not specified, it is automatically derived from the file name. If the filename is passed as `"-"`, the image is read from standard input, in which case the second argument is mandatory.

Optionally, the `--read-only` switch may be used to create a read-only container or VM image. No cryptographic validation is done when importing the images. Much like image downloads, ongoing imports may be listed with `list-transfers` and aborted with `cancel-transfer`.

#### `import-fs DIRECTORY [NAME]`

Imports a container image stored in a local directory into `/var/lib/machines/`, operates similar to `import-tar` or `import-raw`, but the first argument is the source directory. If supported, this command will create `btrfs` snapshot or subvolume for the new image.

#### `export-tar NAME [FILE]`, `export-raw NAME [FILE]`

Exports a TAR or RAW container or VM image and stores it in the specified file. The first parameter should be a VM or container image name. The second parameter should be a file path the TAR or RAW image is written to. If the path ends in `".gz"`, the file is compressed with `gzip`, if it ends in `".xz"`, with `xz`, and if it ends in `".bz2"`, with `bzip2`. If the path ends in neither, the file is left uncompressed. If the second argument is missing, the image is written to standard output. The compression may also be explicitly selected with the `--format=` switch. This is in particular useful if the second parameter is left unspecified.

Much like image downloads and imports, ongoing exports may be listed with `list-transfers` and aborted with `cancel-transfer`.

Note that, currently, only directory and subvolume images may be exported as TAR images, and only raw disk images as RAW images.

#### `list-transfers`

Shows a list of container or VM image downloads, imports and exports that are currently in progress.

#### `cancel-transfer ID...`

Aborts a download, import or export of the container or VM image with the

specified ID. To list ongoing transfers and their IDs, use list-transfers.

## OPTIONS

The following options are understood:

`-p, --property=`

When showing machine or image properties, limit the output to certain properties as specified by the argument. If not specified, all set properties are shown. The argument should be a property name, such as "Name". If specified more than once, all properties with the specified names are shown.

`-a, --all`

When showing machine or image properties, show all properties regardless of whether they are set or not.

When listing VM or container images, do not suppress images beginning in a dot character (".").

When cleaning VM or container images, remove all images, not just hidden ones.

`--value`

When printing properties with show, only print the value, and skip the property name and "=".

`-l, --full`

Do not ellipsize process tree entries or table. This implies

`--max-addresses=full`.

`--kill-who=`

When used with kill, choose which processes to kill. Must be one of leader, or all to select whether to kill only the leader process of the machine or all processes of the machine. If omitted, defaults to all.

`-s, --signal=`

When used with kill, choose which signal to send to selected processes. Must be one of the well-known signal specifiers, such as SIGTERM, SIGINT or SIGSTOP. If omitted, defaults to SIGTERM.

`--uid=`

When used with the shell command, chooses the user ID to open the interactive shell session as. If the argument to the shell command also specifies a user name, this option is ignored. If the name is not specified in either way, "root" will be used by default. Note that this switch is not supported for the

login command (see below).

**-E NAME=VALUE, --setenv=NAME=VALUE**

When used with the shell command, sets an environment variable to pass to the executed shell. Takes an environment variable name and value, separated by "=".

This switch may be used multiple times to set multiple environment variables.

Note that this switch is not supported for the login command (see below).

**--mkdir**

When used with bind, creates the destination file or directory before applying the bind mount. Note that even though the name of this option suggests that it is suitable only for directories, this option also creates the destination file node to mount over if the object to mount is not a directory, but a regular file, device node, socket or FIFO.

**--read-only**

When used with bind, creates a read-only bind mount.

When used with clone, import-raw or import-tar a read-only container or VM image is created.

**-n, --lines=**

When used with status, controls the number of journal lines to show, counting from the most recent ones. Takes a positive integer argument. Defaults to 10.

**-o, --output=**

When used with status, controls the formatting of the journal entries that are shown. For the available choices, see journalctl(1). Defaults to "short".

**--verify=**

When downloading a container or VM image, specify whether the image shall be verified before it is made available. Takes one of "no", "checksum" and "signature". If "no", no verification is done. If "checksum" is specified, the download is checked for integrity after the transfer is complete, but no signatures are verified. If "signature" is specified, the checksum is verified and the image's signature is checked against a local keyring of trustable vendors. It is strongly recommended to set this option to "signature" if the server and protocol support this. Defaults to "signature".

**--force**

When downloading a container or VM image, and a local copy by the specified

local machine name already exists, delete it first and replace it by the newly downloaded image.

`--format=`

When used with the `export-tar` or `export-raw` commands, specifies the compression format to use for the resulting file. Takes one of "uncompressed", "xz", "gzip", "bzip2". By default, the format is determined automatically from the image file name passed.

`--max-addresses=`

When used with the `list-machines` command, limits the number of ip addresses output for every machine. Defaults to 1. All addresses can be requested with "all" as argument to `--max-addresses=`. If the argument to `--max-addresses=` is less than the actual number of addresses, "... follows the last address.

`-q, --quiet`

Suppresses additional informational output while running.

`-H, --host=`

Execute the operation remotely. Specify a hostname, or a username and hostname separated by "@", to connect to. The hostname may optionally be suffixed by a port ssh is listening on, separated by ":", and then a container name, separated by "/", which connects directly to a specific container on the specified host. This will use SSH to talk to the remote machine manager instance. Container names may be enumerated with `machinectl -H HOST`. Put IPv6 addresses in brackets.

`-M, --machine=`

Connect to `systemd-machined.service(8)` running in a local container, to perform the specified operation within the container.

`--no-pager`

Do not pipe output into a pager.

`--no-legend`

Do not print the legend, i.e. column headers and the footer with hints.

`--no-ask-password`

Do not query the user for authentication for privileged operations.

`-h, --help`

Print a short help text and exit.

--version

Print a short version string and exit.

## MACHINE AND IMAGE NAMES

The machinectl tool operates on machines and images whose names must be chosen following strict rules. Machine names must be suitable for use as host names following a conservative subset of DNS and UNIX/Linux semantics. Specifically, they must consist of one or more non-empty label strings, separated by dots. No leading or trailing dots are allowed. No sequences of multiple dots are allowed. The label strings may only consist of alphanumeric characters as well as the dash and underscore. The maximum length of a machine name is 64 characters.

A special machine with the name ".host" refers to the running host system itself. This is useful for execution operations or inspecting the host system as well. Note that machinectl list will not show this special machine unless the --all switch is specified.

Requirements on image names are less strict, however, they must be valid UTF-8, must be suitable as file names (hence not be the single or double dot, and not include a slash), and may not contain control characters. Since many operations search for an image by the name of a requested machine, it is recommended to name images in the same strict fashion as machines.

A special image with the name ".host" refers to the image of the running host system. It hence conceptually maps to the special ".host" machine name described above. Note that machinectl list-images will not show this special image either, unless --all is specified.

## FILES AND DIRECTORIES

Machine images are preferably stored in /var/lib/machines/, but are also searched for in /usr/local/lib/machines/ and /usr/lib/machines/. For compatibility reasons, the directory /var/lib/container/ is searched, too. Note that images stored below /usr are always considered read-only. It is possible to symlink machines images from other directories into /var/lib/machines/ to make them available for control with machinectl.

Note that some image operations are only supported, efficient or atomic on btrfs file systems.

Disk images are understood by systemd-nspawn(1) and machinectl in three formats:

- ? A simple directory tree, containing the files and directories of the container to boot.
- ? Subvolumes (on btrfs file systems), which are similar to the simple directories, described above. However, they have additional benefits, such as efficient cloning and quota reporting.
- ? "Raw" disk images, i.e. binary images of disks with a GPT or MBR partition table. Images of this type are regular files with the suffix ".raw".

See `systemd-nspawn(1)` for more information on image formats, in particular its `--directory=` and `--image=` options.

## EXAMPLES

Example 1. Download an Ubuntu image and open a shell in it

```
# machinectl pull-tar https://cloud-images.ubuntu.com/trusty/current/trusty-server-cloudimg-amd64-root.tar.gz
# systemd-nspawn -M trusty-server-cloudimg-amd64-root
```

This downloads and verifies the specified .tar image, and then uses `systemd-nspawn(1)` to open a shell in it.

Example 2. Download a Fedora image, set a root password in it, start it as service

```
# machinectl pull-raw --verify=no
https://dl.fedoraproject.org/pub/fedora/linux/releases/27/CloudImages/x86_64/images/Fedora-Cloud-Base-27-1.6.x86_64.raw.xz
```

```
# systemd-nspawn -M Fedora-Cloud-Base-27-1.6.x86_64
# passwd
# exit
# machinectl start Fedora-Cloud-Base-27-1.6.x86_64
# machinectl login Fedora-Cloud-Base-27-1.6.x86_64
```

This downloads the specified .raw image with verification disabled. Then, a shell is opened in it and a root password is set. Afterwards the shell is left, and the machine started as system service. With the last command a login prompt into the container is requested.

Example 3. Exports a container image as tar file

```
# machinectl export-tar fedora myfedora.tar.xz
```

Exports the container "fedora" as an xz-compressed tar file `myfedora.tar.xz` into the current directory.

Example 4. Create a new shell session

```
# machinectl shell --uid=lennart
```

This creates a new shell session on the local host for the user ID "lennart", in a su(1)-like fashion.

## EXIT STATUS

On success, 0 is returned, a non-zero failure code otherwise.

## ENVIRONMENT

### `$$SYSTEMD_PAGER`

Pager to use when `--no-pager` is not given; overrides `$PAGER`. If neither `$$SYSTEMD_PAGER` nor `$PAGER` are set, a set of well-known pager implementations are tried in turn, including `less(1)` and `more(1)`, until one is found. If no pager implementation is discovered no pager is invoked. Setting this environment variable to an empty string or the value "cat" is equivalent to passing `--no-pager`.

### `$$SYSTEMD_LESS`

Override the options passed to `less` (by default "FRSXMK").

Users might want to change two options in particular:

#### K

This option instructs the pager to exit immediately when Ctrl+C is pressed.

To allow `less` to handle Ctrl+C itself to switch back to the pager command prompt, unset this option.

If the value of `$$SYSTEMD_LESS` does not include "K", and the pager that is invoked is `less`, Ctrl+C will be ignored by the executable, and needs to be handled by the pager.

#### X

This option instructs the pager to not send termcap initialization and deinitialization strings to the terminal. It is set by default to allow command output to remain visible in the terminal even after the pager exits. Nevertheless, this prevents some pager functionality from working, in particular paged output cannot be scrolled with the mouse.

See `less(1)` for more discussion.

### `$$SYSTEMD_LESSCHARSET`

Override the charset passed to `less` (by default "utf-8", if the invoking terminal is determined to be UTF-8 compatible).

## `$SYSTEMD_COLORS`

The value must be a boolean. Controls whether colorized output should be generated. This can be specified to override the decision that `systemd` makes based on `$TERM` and what the console is connected to.

## `$SYSTEMD_URLIFY`

The value must be a boolean. Controls whether clickable links should be generated in the output for terminal emulators supporting this. This can be specified to override the decision that `systemd` makes based on `$TERM` and other conditions.

## SEE ALSO

`systemd(1)`, `systemd-machined.service(8)`, `systemd-nspawn(1)`, `systemd.special(7)`,  
`tar(1)`, `xz(1)`, `gzip(1)`, `bzip2(1)`

`systemd` 245

`MACHINECTL(1)`