



How to set path for sudo commands

Asked 8 years, 1 month ago Modified 5 months ago Viewed 91k times



If I issue

74

`sudo my-command`



how does Linux look for that `my-command` ?



The `my-command` is in my PATH. I can invoke it without any problem. However, when I invoke it with `sudo`, I'll get `command not found`. How to overcome it?

EDIT: That "*Possible duplicate*"'s selected answer is wrong, well, at least not to the point. This answer, from terdon, is the correct one.

linux

ubuntu

command-line

path

sudo

Share Improve this question

edited Jun 9, 2021 at 18:56

asked Jun 13, 2015 at 16:43

Follow



xpt

8,191

38

100

150

Sorted by:

Highest score (default)



4 Answers



This is normally set by the `secure_path` option in `/etc/sudoers`. From `man sudoers`:

103

`secure_path` Path used for every command run from `sudo`. If you don't trust the people running `sudo` to have a sane PATH environment variable you may want to use this. Another use is if you want to have the "root path" be separate from the "user path". Users in the group specified by the `exempt_group` option are not affected by `secure_path`. This option is not set by default.



To run commands that are not in the default `$PATH`, you can either

1. Use the full path: `sudo ~/bin/my-command`; or
2. Add the directory containing the command to `secure_path`. Run `sudo visudo` and edit the secure path line:

Defaults

secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/home/

Save the file and next time you run `sudo` , the directory `~/bin` will be in its `$PATH` .

Share Improve this answer Follow

answered Jun 13, 2015 at 22:25



[terdon on strike](#)

52.4k 14 124 168

- 4 Or just comment out the whole line if this isn't a production machine and we don't care. Then it will use the users' PATH. It says that it's not set by default, but that may not always be true... – [Nagev](#) Jan 30, 2019 at 10:11



This is what I used for a workaround:

7

```
sudo cp $(which my-command) /usr/bin
...
```



The `which` command is executed in a subshell that is non-root, so it is able to find `my-command` , then, `sudo` copies the executable to a path that the `root` user can access. Not great for security, but it was ok for me running a docker image that was being destroyed right after the command was run.

Share Improve this answer Follow

answered Sep 18, 2018 at 1:05



[Brenden Smith](#)

Brenc 71 1 2

I prefer this answer. It also worked like charm, Thanks! – [Davlet D](#) Nov 8, 2020 at 15:49



6

You can also use `sudo env PATH=$PATH ...rest_of_command_here...` . Since that's not so convenient to type, I've added an alias `alias sudop='sudo env PATH=$PATH'` . The `sudop` (rather than aliasing `sudo` itself) is a reminder to me to make me aware that I am preserving my current environment path.



Share Improve this answer Follow

answered Jun 9, 2021 at 2:43



[Martin_W](#)

181 1 2



0

How to symlink or copy an executable to a bin dir that `sudo` has access to

I'm using a text editor called `micro` . I ran `sudo micro /etc/sysctl.conf` and got this error:



```
sudo: micro: command not found
```



That's because `micro` is in my `~/bin` dir, but that's a personal bin dir to my user only. To give `sudo` access to it, I can either symlink it or copy it to `/usr/bin` as well:

```
# symlink it (my preference)
# Create a symlink at /usr/bin/micro which points to my personal ~/bin/micro
# executable
sudo ln -si ~/bin/micro /usr/bin

# OR: copy it
sudo cp -i ~/bin/micro /usr/bin
```

Done. Now when I run `sudo micro` it works fine.

Share Improve this answer Follow

answered Jan 25 at 22:07



Gabriel Staples

1,758

1

22

34