



Unix & Linux Stack Exchange is a question and answer site for users of Linux, FreeBSD and other Un*x-like operating systems. Join them; it only takes a minute:

Sign up


Here's how it works:



Anybody can ask a question



Anybody can answer



The best answers are voted up and rise to the top

SSH and home directory permissions

It took me hours to solve this SSH problem with one of my class accounts on my school's servers.


I couldn't ssh into one particular class account without entering my password, while passwordless authentication worked with my other class accounts. The `.ssh/` directory and all of its contents had the same, correct permissions as the other class accounts.

Turns out the problem was the permissions set on my own home directory. Passwordless authentication did not work when the permissions on my HOME directory were set to 770 (regardless of the permissions set for `.ssh/`), but it worked with permissions set to 755 or 700.

Anyone know why SSH does this? Is it because the home directory permissions are too permissive? Why does SSH refuse to authenticate with the public/private keys when the home directory is set more permissive than 700?

/ ssh / permissions


edited Apr 23 '12 at 23:32



Gilles

445k 91 837 1334

asked Apr 23 '12 at 20:23



action_potato

651 2 8 7

1

Confirmed the answers below; the problem was the group permissions on the home folder were incorrectly set (the error message from `auth.log` was: 'Authentication refused: bad ownership or modes for directory /home/<user>'). I see SSH is right to be picky about home dir permissions. – action_potato Apr 23 '12 at 20:54

5

Did you know about the tag wikis we have here? If you click on `ssh` and then `Learn more`, you'll see a checklist for what to do when SSH isn't working, and it mentions the home directory permissions. – Gilles Apr 23 '12 at 23:34

Ah, sorry I didn't know about that! Thanks for the heads up. – action_potato Apr 24 '12 at 0:03


2 Answers

This is the default behavior for SSH. It protects user keys by enforcing `rwX-----` on `$HOME/.ssh` and ensuring only the owner has write permissions to `$HOME`. If a user other than the respective owner has write permission on the `$HOME` directory, they could maliciously modify the permissions on `$HOME/.ssh`, potentially hijacking the user keys, `known_hosts`, or something similar. In summary, the following permissions on `$HOME` will be sufficient for SSH to work.

- `rwX-----`
- `rwXr-X---`
- `rwXr-Xr-X`

SSH will not work correctly and will send warnings to the log facilities if any variation of `g+w` or `o+w` exists on the `$HOME` directory. However, the administrator can override this behavior by defining `StrictModes no` in the `sshd_config` (or similar) configuration file, though it should be clear that this is *not recommended*.

edited May 1 '12 at 21:02



George M

7,856 1 30 45


answered Apr 23 '12 at 20:31

Thanks for mentioning `StrictModes no`. In my setup, an ACL is configured on the target user's home directory and all descendants to allow modifications by a semiprivileged user (`u:operator:rwX`), and SSH didn't like this. – intelix Jul 25 '15 at 2:06

77x on your home directory means that everybody with correct GID can move your `.ssh` directory and replace it with another one. Users with the correct GID have `write/exec` permissions on the home directory and therefore can rename/create files/directories.

SSH is very picky when it comes to permissions, and it should.

answered Apr 23 '12 at 20:50



jippie

7,791 5 27 53

