

SSHswitch(1) u.blanxd.sshswitch32 | version 1.0

Blanxd.H <blanxd.h@gmail.com> (reddit.com/u/blanxd)

Nov 2020

NAME

SSHswitch - an iOS (jailbroken) setuid tool for configuring the OpenSSH server, and starting/stopping it.

SYNOPSIS

SSHswitch

```
SSHswitch on [port(s) [0|no|1|yes|- [0|no|1|yes|- [0|no|1|yes|- [1|0|-]]]]]
SSHswitch off [port(s) [0|no|1|yes|- [0|no|1|yes|- [0|no|1|yes|- [1|0|-]]]]]
SSHswitch s | status
SSHswitch p | port [awut]
SSHswitch a | allowRoot
SSHswitch w | withPassword
SSHswitch u | usingKey
SSHswitch t | twoFactor
SSHswitch i | info | ij
SSHswitch l | letEmLinger
SSHswitch k | kickEm
SSHswitch e | evenIfLocked
SSHswitch f | forbidIfLocked
SSHswitch r | runOnBoot
SSHswitch n | notOnBoot
SSHswitch b | bootAsToggled
SSHswitch d | defaultState
SSHswitch v | version
SSHswitch -h
SSHswitch h | help | --help
```

DESCRIPTION

This tool is used internally by the GUI tweaks

- “OpenSSH Settings (32-bit)” (u.blanxd.OpenSshPort32)
- “OpenSSH Flipswitch (32-bit)” (u.blanxd.OpenSshFlipswitch32)

It sets OpenSSH options in /etc/ssh/sshd_config and /Library/LaunchDaemons/com.openssh.sshd.plist, it uses launchctl to start/stop the server and read its running status, and it also keeps a few settings files of its own (and the GUI toggle tools) in /etc/sshd/. These settings toggle

- whether it kicks ssh sessions off the server (the iDevice) when it turns the server off.
- whether using the Flipswitch toggle is allowed while the device is locked.
- how it uses launchctl for starting/stopping the server (ie. whether ON or OFF are only temporary/runtime or permanent).

com.openssh.sshd.plist is being read and written to as a plain text XML file, so if anyone (any OpenSSH packager) ever makes it a binary or JSON plist then this tool might not be able to change nor report the correct port numbers

(but it wouldn't corrupt it either).

The SSH server itself runs as *root*, so it wouldn't be possible for the *mobile* user (the iOS GUI and apps) to start/stop this, that's why this tool uses *setuid 0*, and is installed as such, so it can be used by any user on the system.

When it successfully turns the server on or off, it emits a system-wide Darwin notification, which may be observed by any process. The name of the notification is "com.openssh.sshd/on" or "com.openssh.sshd/off". If it's an internal sshd restart after changing some settings then there will be two of those, there are no exceptions.

All subcommands, although representing English words/expressions, are really only read from the beginning until a unique instruction can be determined, which mostly means the 1st letter. Two characters are needed for *on*, *off*, *ij*, *-h*, *--help*.

SUBCOMMANDS (Info requests)

SSHswitch

calling it without any args/subcommands simply exits with the return code 0 if OpenSSH is running, 1 if it isn't.

status | s

prints whether SSH is **on** or **off**

port | p [awut]

prints the port(s) OpenSSH is currently configured to listen on, each one on a separate line.

It can be followed by any or all of the 4 letters described next (in any order), in which case it also prints 'yes' or 'no' preceded by the corresponding letter, each on a separate line, and in this case the exit code reflects the running status of the server. This is the API output that OpenSSH Settings uses. Neither the ports nor the yes's/no's are guaranteed to be in any particular order.

allowRoot | a

prints whether root is allowed to log in (**yes** or **no**)

withPassword | w

prints whether it's allowed to login with a password (**yes** or **no**)

usingKey | u

prints whether it's allowed to login with a key (**yes** or **no**)

twoFactor | t

prints whether both the password and a key are required (**yes** or **no**)

info | i | ij

list active sessions, return 1 if there aren't any. Adding 'j' makes it output the info in JSON format, without a newline.

defaultState | d

prints the startup state: BootAsToggled | RunOnBoot | NotOnBoot.

version | v

prints the version of this tool.

-h

prints a short list of all the possible subcommands.

help | h | --help

prints a more comprehensive description (still less than in this manpage).

SUBCOMMANDS (OpenSSH settings)

All configuration concerning OpenSSH itself happens with the *on* or *off* subcommands. The options that follow the *on/off* determine if anything gets written to `/etc/ssh/sshd_config` and/or `/Library/LaunchDaemons/com.openssh.sshd.plist`, and whether it starts/stops/restarts the server service. Each can be either *1*, *0*, *yes*, *no*, or *-* (a minus, which skips that option if a next one needs to get set). The order of these options needs to be given as follows, ie. the only way to skip any (from the beginning) is to use a minus.

on | off

determines if the server stays on/off after the command has finished, regardless if any additional options follow.

1. port[,port[...]]-

numeric port number(s) that the server needs to listen on. This tool only saves ports 22 (the standard) or anything from 1001 through 65535, there is no error indication if others are requested, but they don't get saved. This is just an arbitrary precaution I took so noone would make it listen on some port used by other known services (although yes, there are bigger known ports as well (TODO maybe)). Up to 8 ports may be specified, separated by commas (no spaces). The *port* subcommand reads the result.

2. 1|0|yes|no|-

determines if root is allowed to log in (PermitRootLogin keyword in `sshd_config`). The *allowRoot* subcommand reads the result.

3. 1|0|yes|no|-

determines if it's allowed to log in using a password (PasswordAuthentication keyword in `sshd_config`). The *withPassword* subcommand reads the result.

4. 1|0|yes|no|-

determines if it's allowed to log in using a public key (PubkeyAuthentication keyword in `sshd_config`). The *usingKey* subcommand reads the result.

5. 1|0|-

the last '1' or '0' (if given) is not an OpenSSH setting. This makes us kick (or not) all possible (incoming) ssh/sftp/scp sessions off the server, even if the subcommand is *on*. This option also overrides the *letEmLinger/kickEm* setting (see below). This one I cannot guarantee to remain the 5th in case any functionality gets added in the future, it'll just always be the last one.

If both the 3rd and 4th options are given 0|no then this doesn't make much sense, either Password or PubKey authentication should be allowed, else it's a lockout (unless using some more advanced authentication schemes, which are possible, but I'm not sure if anyone uses those in iOS). So this tool instead reverts these to 'yes', and additionally makes the server require both, *twoFactor*. Ie. a user must have their pubkey in the `authorized_keys` file, and in addition they must provide the password when logging in. Ie. Extra Secure :)

Changing these settings may make the tool restart the server if it's issued while it's running. Definitely if the port(s) get altered, but also for other settings if `sshd` isn't using `launchd` to listen and runs as a daemon on its own (a few jailbreaks have it like that).

These settings do not persist when the OpenSSH server gets re-installed or upgraded via a package manager, because the OpenSSH server package simply overwrites `/etc/ssh/sshd_config` and `/Library/LaunchDaemons/com.openssh.sshd.plist`, thus discarding any modifications we have made.

SUBCOMMANDS (Tool/Tweak settings)

kickEm | k

makes us kick all possible (incoming) ssh/sftp/scp sessions off the server whenever an *off* subcommand is issued. This is the default when this tweak is installed. Use *letEmLinger* to revert this. Can be overridden with the last (5th) option as *0* to the *off* command.

letEmLinger | l

the opposite of the above *kickEm* subcommand, ie. the *off* subcommand would simply turn off the server and never mind if anyone is logged in. Can be overridden with the last (5th) option as *1* to the *off* command.

forbidIfLocked | f

this is used by the Flipswitch toggle. Makes it have no effect when the device is locked. This is the default for

u.blanxd.OpenSshFlipswitch32. Use *evenIfLocked* to revert this.

evenIfLocked | e

this is used by the Flipswitch toggle. The opposite of the above *forbidIfLocked*, makes it work even when the device is locked.

SUBCOMMANDS (Settings for Launchd)

runOnBoot | r

This is the default when OpenSSH gets installed ie. whenever the device gets re-jailbroken (or just rebooted in case the jb is untethered) then the ssh server starts up as well. Internally, the tool starts the server with ‘launchctl -w’ and stops it with ‘launchctl -F’.

notOnBoot | n

This makes the ssh server not start up when the device is re-jailbroken or rebooted. Internally, the tool starts the server with ‘launchctl -F’ and stops it with ‘launchctl -w’.

bootAsToggled | b

This makes the ssh server either start or not, based on whichever state it was in before the restart/re-jailbreaking. Internally, the tool always starts and stops the server with ‘launchctl -w’.

Changing this setting may make the tool restart the server, depending on which way it got turned to while the server is running. Similarly it might get turned on for a millisecond if changing this while the server is off.

This setup does not persist when the OpenSSH server gets re-installed or upgraded, because usually the postinst script in the OpenSSH server package simply uses ‘launchctl -w’ to load it up after any installation. Just toggling it back and forth once makes it work again.

EXIT STATUS

Any exit status ≥ 10 is an error in any case. You should really always check it.

Regular subcommands should return 0.

Without any subcommands, or in case there are extra letters after *port*, it returns 0 if the server is running, and 1 if it isn’t.

info and *ij* return 1 if there aren’t any sessions.

EXAMPLES

Most info request commands should be easy, they’re just saying *yes* or *no*, and *on* or *off*.

Showing some API type subcommands here.

```
iPhone:~ mobile$ # show the port(s)
iPhone:~ mobile$ SSHswitch port
10222
22
iPhone:~ mobile$ # turn off, save port 22 and allow password auth
iPhone:~ mobile$ SSHswitch off 22 - 1
iPhone:~ mobile$ # report ports and if password auth allowed
iPhone:~ mobile$ SSHswitch p w
22
wyes
iPhone:~ mobile$ # report all the OpenSSH settings this tool supports
iPhone:~ mobile$ SSHswitch port awut
22
ano
```

```
wno
uyes
tno
iPhone:~ mobile$ # exit status shows if it's running (the toggle tweaks use this)
iPhone:~ mobile$ SSHswitch # 1 means it's off
iPhone:~ mobile$ echo $?
1
iPhone:~ mobile$ # turning twoFactor on among other things
iPhone:~ mobile$ SSHswitch on 10222 1 0 0
iPhone:~ mobile$ SSHswitch p awut
10222
ayes
wyes
tyes
uyes
iPhone:~ mobile$ # the previous command returned 0 if it's running
iPhone:~ mobile$ echo $?
0
iPhone:~ mobile$ # kick everyone out without changing anything
iPhone:~ mobile$ SSHswitch on - - - 1
Connection to 10.8.9.14 closed.
```

COPYRIGHT (MIT/Expat)

Copyright (c) 2020-11 Blanxd.H blanxd.h@gmail.com

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.