

Initial Configuration

- [Setting up doas](#)
- [Disable root login?](#)
- [Connect to Wi-Fi on OpenBSD](#)
- [Laptop – Touchpad tap to click](#)

Setting up doas

Enable `doas` for user

It is generally not recommended on any *NIX-based system to login as `root` on a regular basis. Apart from security reasons, not logging directly as `root` also prevents you from making dumb mistakes and causing issues (with a less privileged account, usually less stuff can break). On Linux, this is solved by leveraging the `sudo` program. It allows you to elevate privileges to `root`, set which accounts are allowed to do so, perform actions on behalf of different accounts, restrict accounts to execute only some commands as `root` etc. The amount of options makes `sudo` rather bloated according to some people. To offer a simpler alternative to `sudo`, `doas` was created. `Doas`'s main function is essentially the same as `sudo` - safely elevate privileges, run commands as another user etc. Due to its simplicity and smaller codebase (easier to audit, less room for error), OpenBSD uses the simpler `doas` program.

We currently have two accounts - `root` and the user account you have created during the setup, let's say it's `bob`.

Login as `root` to create configuration file. `Doas` comes preinstalled, but doesn't create the configuration file by default, we have to do it manually.

```
(root)$ touch /etc/doas.conf
```

We are going to allow a special group called `wheel` (which our account `bob` should be part of by default) to execute commands as `root`. To check if your account is in the `wheel` group, use the `groups {user}` command or `groupinfo wheel`

```
$ groups bob
bob wheel
```

```
$ groupinfo wheel
name=wheel
passwd=*
gid=0
members root bob
```

Open `/etc/doas.conf` in your favorite editor and add the following to the first line. For additional commands and information, run `man doas.conf`.

- `permit` - We want to permit the `wheel` group to do certain things, use `deny` to deny

- `nopass` - I have a long password and I'm fine with typing in only to log in, this option makes sure that when I call `doas`, it doesn't ask for a password. To type password every time you want to use `doas`, omit this option. Alternatively, replace with `persist` so it won't ask you for a password for 5 minutes after issuing the first elevated command.
- `:wheel` - Apply the previously mentioned options to the `wheel` group.

```
(root)$ nvim /etc/doas.conf
```

The command above will only work if you have `neovim` installed. If not, use the default `vi`.

```
permit nopass :wheel - /etc/doas.conf
```

To test your configuration file, run `doas -C /etc/doas.conf {command}`, replace `{command}` with anything like `cat`, `vi` etc. This will tell you whether you are allowed to run that specific command as `root`. We should be now able to run all commands as `root`. You may need to log in/log out if it doesn't work at first.

After you are done with the steps above, make sure `/etc/doas.conf` is owned by `root` and group `wheel` and has sane permissions (only writable by `root`). I also like tightening permissions even further, you might **not want to** do the same.

```
doas chmod 400 /etc/doas.conf
```

Disable root login?

Disable root login?

Unlike on Linux, where disabling the `root` account is one of the first things I do after setting up `sudo`, on OpenBSD, I don't find it necessary and it is even advised against. Try locking the `root` account with `usermod` and you will see that it won't even let you. Best practise in this case:

- Give `root` a really long random password.
- Write it down physically, store in a safe place.
- When needed (system recovery), use `root` to fix whatever is broken.

Connect to Wi-Fi on OpenBSD

Even though I prefer physical connection, even on a laptop, and I plug in an Ethernet cable whenever I can, it is sometimes necessary to use WiFi. This requires remembering a few commands on OpenBSD.

Figure out network interface name

OpenBSD offers support for various network cards. It used to be way worse, but now I would say it's pretty uncommon to find a wireless card that doesn't work at all, however, features might differ from driver to driver.

First, we need to get the name of our wireless interface. To do so, run `ifconfig`:

```
$ ifconfig
```

In the output, look for [one of these interface names](#) (list of possible wireless interfaces in OpenBSD, the name differs based on the wireless driver that handles the Wi-Fi card). Other clues that it's a wireless interface include `wlan`, `802.11` etc.

```
iwn0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
lladdr 00:11:22:33:44:55
    index 2 priority 4 llprio 3
    groups: wlan
    media: IEEE802.11 autoselect
    status: no network
    ieee80211: nwid ""
```

In my case, the interface name is `iwn0`.

Up the interface

Before we can start scanning or connecting to networks, we need to make sure the wireless interface is `UP`. Replace `iwn0` with the name of your interface.

```
$ doas ifconfig iwn0 up
```

Scan for Wi-Fi networks

Basically all operating systems allow you to view all available wireless networks around you in some way. Usually in a form of a list that appears after you turn on Wi-Fi access. It's basically the same on OpenBSD, it just took me a bit longer to figure out since OpenBSD won't give you shiny buttons, but a simple (and powerful!) terminal.

```
$ ifconfig iwn0 scan
```

Which will give you the aforementioned list:

```
iwn0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
lladdr 00:11:22:33:44:55
    index 2 priority 4 llprio 3
    groups: wlan
    media: IEEE802.11 autoselect
    status: no network
    ieee80211: nwid ""
    nwid MyWiFi chan 11 bssid 55:44:33:22:55:66 -49dBm HT-MCS15
privacy,short_preamble,short_slottime,wpa2
    nwid AnotherNetwork chan 112 bssid 88:00:11:99:55:33 -58dBm HT-MCS15
privacy,short_preamble,short_slottime,wpa2
    nwid TP-Link chan 7 bssid 99:55:11:11:11:11 -91dBm HT-MCS15
privacy,short_slottime,radio_measurement
lll...
...
```

Connect to a Wifi Network

Choose your desired network from the list generated by `ifconfig iwn0 scan` and run:

```
$ doas ifconfig iwn0 nwid "MyWiFi" wpakey MySuperSecretPassword
```

- `iwn0` – Name of the interface
- `MyWiFi` – Name of your Wi-Fi network
- `MySuperSecretPassword` – password for your network, omit `wpakey` if the network is open

You should now be connected, but you most likely won't be able to ping anything yet.

```
$ ping 8.8.8.8
```

```
PING 8.8.8.8 (8.8.8.8): 56 data bytes
```

```
ping: sendmsg: No route to host
```

```
ping wrote 8.8.8.8 64 chars, ret=-1
```

That is because **you haven't asked for a DHCP lease yet.**

Get IP from DHCP server

This option assumes you want a DHCP address, not a static one. To get the address from DHCP, simply run `dhclient` on your Wi-Fi interface.

```
$ doas dhclient iwn0
```

```
iwn0: 192.168.1.58 lease accepted from 192.168.1.1 (11:11:11:11:11:11)
```

Success! You are now connected to a Wi-Fi network with a valid IP address. There's, however, way more to discover and learn – auto connecting, configuration files so you don't have to remember all the commands etc. This guide was mainly designed to show you how to connect to a wireless network one-off.

Laptop – Touchpad tap to click

If you have installed OpenBSD on a laptop, you probably want the touchpad to work right. I expect to be able to click when I tap on the touchpad instead of having to press one of the buttons below it. To enable this feature, use `wsconsctl`.

To list current settings, type `doas wsconsctl`, which will show you all current settings controlled by `wsconsctl`.

```
keyboard.type=pc-xt
keyboard.bell.pitch=400
keyboard.bell.period=100
...
...
...
...
mouse.type=synaptics
mouse.rawmode=0
...
...
```

We need to create a `wsconsctl` configuration file, so that the settings persist across reboots. There's an example configuration file located in `/etc/examples/wsconsctl.conf`. Conveniently, touchpad tapping is already as one of the examples.

```
# wscons configurable parameters
#
#keyboard.repeat.dell=200 # change keyboard repeat/delay
...
...
...
#mouse.tp.tapping=1 # enable tapping with default button mappings;
##### for customized mappings, list 3 button codes
```

You can either copy this file to `/etc/wsconsctl.conf` and remove the lines you don't want there or create `/etc/wsconsctl.conf` from scratch and only add the mouse tapping line:


```
$ doas touch /etc/wsconsctl.conf
```

```
/etc/wsconsctl.conf
```

```
mouse.tp.tapping=1
```

Now reboot and it should be enabled.