How To: List CPU Information From The Terminal (lscpu)

In the last article I wrote, I explained how to use ‘lshw’ to get information about your hardware from the terminal. This article will show you how to gather CPU information from the terminal.

This will be yet another quick article, but again this is a valuable tool to do hardware diagnostics when you don’t have other tools available.

Just like last time, the command is self-evident once you really look at it. The command is ‘lscpu‘ and it does what you’d expect – ‘list CPU information’. It is defined as thus in the manual:

> *lscpu – display information about the CPU architecture*

So, let’s give it a shot. Crack open your default terminal emulator by pressing CTRL + ALT + T. And, let’s just start by entering:

[code]lscpu[/code]

And, there you have some fairly easy to understand. It’s often used as a quick way to tell if your CPU supports 64 bit instructions. You’ll see something like this if it does:

[code]Architecture: x86_64[/code]

NOTE: Pretty much all modern computer hardware supports the 64 bit instruction set, but there are still some 32 bit machines out there and being used.

Unlike the ‘lshw’ command, this one doesn’t need to be run as an administrator. It runs as a regular user just fine and

running it as an admin doesn’t get you any more information.

Also unlike the ‘lshw’ command, there isn’t any other way to run it that’s all that interesting. You can read the man page, but you’ll seldom need to use this in any other way other than the way described here. Just type the command, get the information, and move on with whatever it was you were doing!

See? I told you this one would be another quick and easy article. Thanks for reading and don’t be scared of signing up to the newsletter. It’s not like you’ll be inundated with piles of unwanted email! You’ll just get notified when there’s a new article and I promise to not sell your email address to anyone.

How To: List Hardware From The Terminal (lshw)

You may not always have inxi available. The person trying to help you may want more specific info. You may need more information about your hardware. Who knows? There’s all sorts of reasons to use ‘lshw’ in your day-to-day computing.

Today is another short and easy article (you’re welcome). It’s just a very brief command, one of several, that we’re going to learn how to use. The command in question is ‘lshw‘. The manual helpfully defines it as:

> *lshw – list hardware*

Which, now that you look at it, makes perfectly good sense.

This is one of those Linux commands that’s not at all cryptic when you think about it. It looks like what it does, it does what you’d intuitively think it does. (It’s actually a subset of the info you get with ‘hwinfo’, but that’s not important right now!)

What is important is that there are two realistic ways to use it – and both of them should be run as an administrator. The first is just as you’d expect:

[code]sudo lshw[/code]

That will output a ton of information about the hardware you have in (or connected to) your computer. It’s rather overwhelming and it’s not often that you’ll be asked to post all of it. You may be asked to (or want to) run it with the -C option, such as:

[code]sudo lshw -C cpu[/code]

You may even be asked to use grep with it, such as:

[code]sudo lshw | grep wireless[/code]

Then, there’s one more way to run the program (under normal circumstances) and that’s to run the whole thing and to get the whole output in a shorter format. The command is probably just like you’d expect.

[code]sudo lshw -short[/code]

Lo and behold! Would you look at that! Ha! Isn’t that fantastic? You actually get readable output that’s suitable for your normal user who just wants to know what sorta hardware they’re working with while getting a few extra bits of info! It even includes juicy nuggets like:

[code]/1 wlxe4beed0e5f5c network Wireless interface[/code]

Now, if you want to refine it even further, why not try this:

[code]sudo lshw -short | grep network[/code]

And, there you have it… You have another way to see your hardware in the terminal and it should be consistent across any major distro you touch. You shouldn't have any trouble using the command and it's easy to remember when you want to 'list the hardware'. There are other uses, but those two are the most common. For more information about the lshw command try:

[code]man lshw[/code]

or

[code]info lshw[/code]

Like always, thanks for reading. Feel free to subscribe to the newsletter. I promise to not give away your private information – and you never know what sorta incentives I may stick in there. See? I told you this one would be short and easy!

How To: Time a Command

Have you ever wanted to know how long it takes to complete a command that you entered in the Linux terminal? Well, wonder no more!

This is going to just be a pretty quick article and easy to follow. There's not a whole lot to explain and it's pretty straightforward. Like often, let's crack open your default terminal by pressing CTRL + ALT + T on your keyboard.

Now, let's take the command:

[code]ls -la[/code]

Unless you have a lot of files, that completes pretty quickly. But, how fast does it really take? Well, simply add the ‘time‘ command before it. Time is simply described in the man page as:

> *time – run programs and summarize system resource usage*

And, for today, it’s going to be pretty easy to use that command. To find out how long it took to list all the files and folders in a directory, you could use:

[code]time ls -la[/code]

Note how it tells you the time beneath the results and, if you want to try something bigger, you can take a look at this command:

[code]for i in {0..99999}; do echo “I love LinuxTips!”; done[/code]

That should take a just a little more time, but you can actually see how long it really took by adding ‘time’ in front of it. So:

[code]time for i in {0..99999}; do echo “I love LinuxTips!”; done[/code]

The output at the end is something like this:

> *real 0m0.566s*
> *user 0m0.423s*
> *sys 0m0.143s*

The ‘real’ is how much time it really took. The ‘user’ is how much time it took for the user. The ‘sys’ is how much time it took for the system – the amount of time that the kernel actually devoted to it.

So, there you have it! You can use the time command to find out how long it takes to run stuff in your terminal. If you're playing with scripting and you're looking to optimize it, this is a valuable tool. If you're just a bit curious, then you now have a new tool.

I told you that it'd be quick and easy! Like always, thanks for reading. Feel free to sign up for the newsletter. I promise to not spam you or sell your email address.

Let's Spin up a Quick Python Server!

Today we're going to use a basic Python server to share files between multiple computers both quickly and easily, by using tools you likely have by default.

First, let's set the stage…

You're eight beers deep while sitting on your couch and playing with virtual machines. You have already downloaded the latest and greatest distro – but it's way over on your desktop and you're both lazy and not too sure about your walking ability!

What to do? What to do?

Sure, you could just download the file all over again – but you're aware of how much the project pays for bandwidth and you're inebriated enough so that if you don't do it right now then it just might not get done! So, you need that file and you need it with a quickness!

I'm going to make a huge assumption here. Someday, I'll write

an article explaining how, but for now we're going to assume that you have SSH (secure shell) enabled on your desktop (in this scenario) and that you know how to use it. So, it's with a giant assumption and a leap of faith when I say that you've successfully used SSH to get to your desktop and you've already navigated to the directory where this latest and greatest distro image resides.

NOTE: If you don't have SSH enabled, surely you have access to a search engine. Go figure it out! Eventually I'll write an article about it, but there are already hundreds of tutorials already out there.

Anyhow, sure… You've used SSH and now you could transfer the file with SCP (secure copy protocol) if you wanted. That's all well and good, but darn it we're aiming for the most contrived situation possible just so I can tell you how to spin up a server with Python! So, for whatever reason, you're hellbent on doing this in your browser. And do this in your browser you shall!

Yeah, there are a ton of ways to transfer files – many (perhaps all) of them better than this! Though you could also use this for testing your HTML and CSS skills. So, there's that! By the way, if my contrived scenario isn't good enough for you, you can make up your own reasons to do this. If your ideas are any good, you can even contribute to the site!

Anyhow, you're now connected to that desktop with SSH and you're in the directory where you store those important files. Now that you're there, run the following command:

[code]python -V[/code]

If you're using Python 1x or 2.x you should probably update, but the magic server command is this:

[code]python -m SimpleHTTPServer[/code]

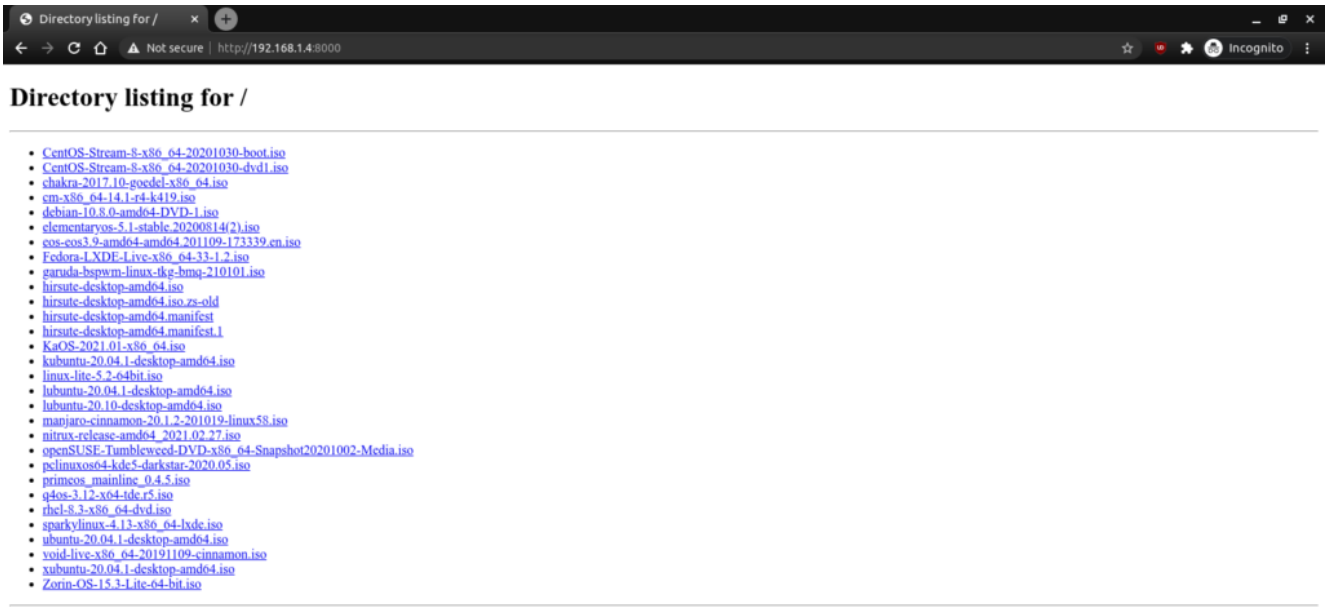
If you’re using Python 3.x then the command will be:

[code]python3 -m http.server[/code]

For the record, the ‘-m’ is telling Python which module to load. Either way, you can now open your browser and enter this tidbit in the address bar:

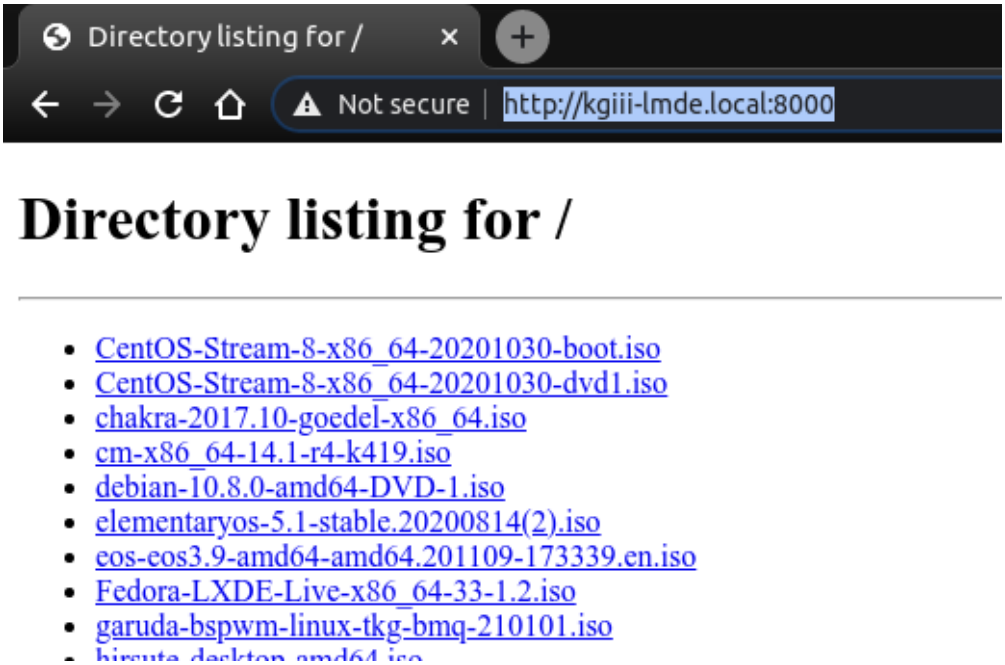
[code]ip.address.of.desktop:8000[/code]

Obviously you should adjust it accordingly. If you’ve done it right, it should look similar to this:



Obviously it will not look the same for you. It’ll have your files!

If you want, you can also connect via the hostname. In the above example, the computer I was connected to is known as ‘kgiii-lmde’ and you need only add the .local for this to work. See the image below:

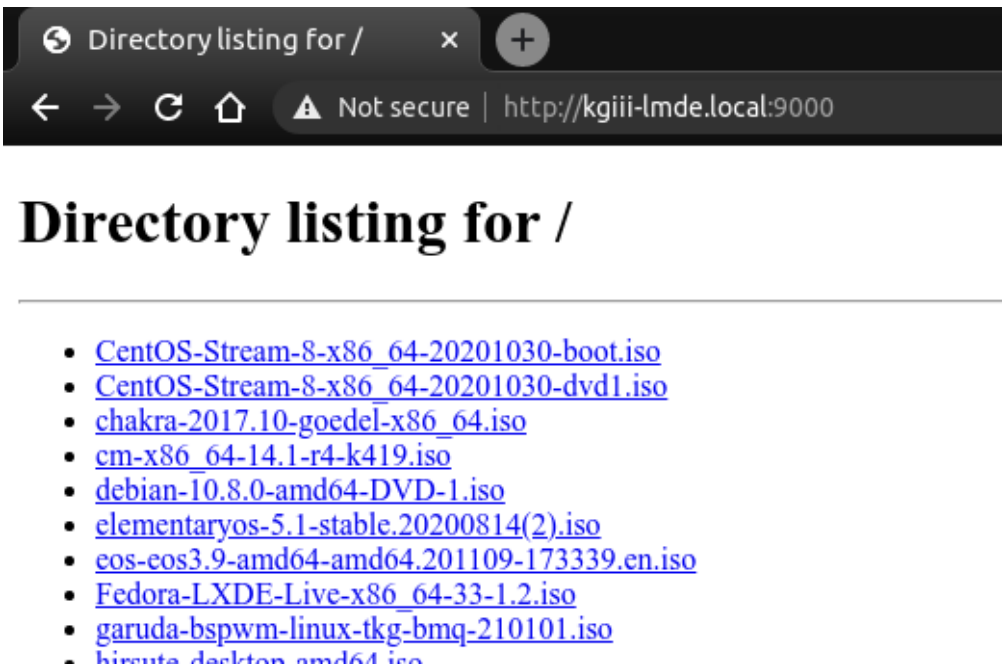


See? No IP address required! You can also use this for the above mentioned SSH!

Now, if you want to do so, you can also change the port number. This is the same for both commands. In both cases, just add your chosen port number at the end. Like so:

[code]python3 -m http.server 9000[/code]

And, again, it should look a bit like this:



Note the changed port number. You should probably avoid reserved ports.

Anyhow, you can use this for all sorts of things. You can now navigate the user's files with a web browser. If you're really insane, you can make this available over the internet by enabling port forwarding – but I'm gonna suggest you not do that. If you're going to do that, you should probably use something more robust and with much finer permission controls. It'd also take quite a bit of work to turn this into a server that'd do anything more than offer up static files. I suspect (but don't know) that it's about as secure as a screen door, so putting this on the public web would just be silly. Don't do that.

Though my written use-case was very contrived, I find myself using this easy server fairly regularly. I actually prefer to use FTP to move files between computers, but I'll spin up a quick Python server so that I can grab things like config files and whatnot. Believe it or not, even wget works. In this instance, if I wanted to grab that Debian ISO, I could just do this:

[code]wget http://kgiii-lmde.local:9000/debian-10.8.0-amd64-DVD-1.iso[/code]

So long as the server is up and running, wget works. Granted, there's not too many contrived situations where I'd need to use wget on top of all this – but it's fun to play with and educational.

What sort of uses can you come up with? Feel free to leave them as comments below. Who knows? Maybe they'll end up being motivations for future articles?

As always, thanks for reading. Don't forget to sign up for the newsletter. Chances are good that I'll not always be on the forums and this is a way for you to keep track of what gets published. I also won't send you any spam, nor will I trade/sell/give your email address to anyone.

How To: Enable Numlock on Modern Lubuntu

At the time of writing, Lubuntu 18.04 is soon to be no longer supported. With the change to LXQt, many of the old tweaks no longer work and you'll need to learn some new ways to do things. In this article, I'll explain how to put numlock into its on position automatically with Lubuntu 20.04 and up.

First, we're going to install 'numlockx' and that's pretty easy. Start by using your keyboard to open the terminal with CTRL + ALT + T. (I really like that keyboard indicating layout feature!)

Now that you have that open, try the following:

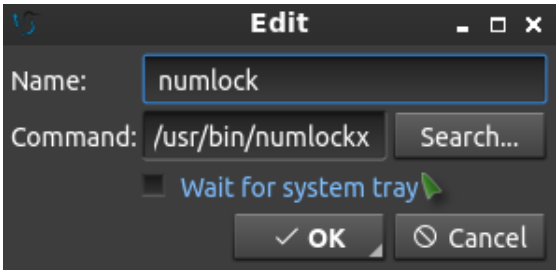
[code]sudo apt install numlockx[/code]

Next, open your applications menu, click on Preferences, click on LXQt settings, and then click on Session settings.

Once you have that open, click on Autostart (on the left) and then Add (on the right). Once that is open, give our numlock a name and enter the following command:

[code]/usr/bin/numlockx[/code]

(Don't forget to click Ok to save it.) It should look a little like this:



See? This is a nice easy one!

And, that’s it. Reboot and it should work. I can’t really drag this out to making it a longer article. Sure, there are other ways to add something to autostart, but I see no reason to do this particular exercise any other way. Why add complexity when this just works out of the box?

Like always, thanks for reading. Don’t forget to sign up for the newsletter. Chances are that I’ll eventually cease posting at all the various forums (when the pandemic is over) so this will help you keep up with the site and keep in touch. Don’t worry, I won’t send you any spam.