The dropbear suite provides both an ssh server and a client application (dbclient), and represents a light alternative to OpenSSH. Since it has a small footprint and uses system resources very well, it is generally used on embed devices, with limited memory and processing power (e.g routers or embed devices), where optimization is a key factor. It provides a lot features, like, for example, X11 forwarding, and it is fully compatible with the OpenSSH public key authentication. In this tutorial we will see how to install it and configure it on Linux.

**In this tutorial you will learn:**

* How to install and configure dropbear on linux
* How to use the dropbearkey, dropbearconvert and dbclient utilities

[](https://linuxconfig.org/images/dropbear_banner.png)

**Software Requirements and Conventions Used**

| *Software Requirements and Linux Command Line Conventions* | |
| --- | --- |
| **Category** | **Requirements, Conventions or Software Version Used** |
| **System** | Distribution-independent (configuration may vary) |
| **Software** | No additional software is needed to follow this tutorial apart from dropbear (see installation instructions below) |
| **Other** | * Familiarity with the command line interface * Basic ssh concepts * Knowledge of systemd |
| **Conventions** | **#** - requires given [linux commands](https://linuxconfig.org/linux-commands" \t "_blank) to be executed with root privileges either directly as a root user or by use of sudo command **$** - requires given [linux commands](https://linuxconfig.org/linux-commands" \t "_blank) to be executed as a regular non-privileged user |

**Installation**

Installing dropbear is a very simple task, since it is available in all the major Linux distributions. All we have to do is to use our favorite distribution package manager. On Debian and its derivatives like Ubuntu, for example, we can use apt:

$ sudo apt install dropbear

On recent versions of fedora we can use the dnf package manager:

$ sudo dnf install dropbear

Dropbear is available in the "community" repository on Archlinux, se we can install it via pacman:

$ sudo pacman -S dropbear

***SUBSCRIBE TO NEWSLETTER*** *Subscribe to Linux Career*[*NEWSLETTER*](https://bit.ly/2X5D30q)*and receive latest Linux news, jobs, career advice and tutorials.*

It is also possible to install the dropbear package on Red Hat Enterprise Linux 7 and CentOS 7, by adding the Epel additional repository and then use the yum package manager:

$ sudo yum install dropbear

Unfortunately, although a version of the Epel repository dedicated to the latest version of RHEL (8) has been already released, it doesn't contain the dropbear package yet. It is still possible to install Epel 7 on Rhel 8, but it must be done with caution.

**Configuring dropbear**

The dropbear service doesn't read its configuration from a dedicated file like OpenSSH. We simply modify the behavior of the program by launching it with the appropriate command line options. How we specify the options depends on the distribution we are using.

On Ubuntu, for example, we modify the /etc/default/dropbear file. Here is its content:

# the TCP port that Dropbear listens on

DROPBEAR\_PORT=22

# any additional arguments for Dropbear

DROPBEAR\_EXTRA\_ARGS=

# specify an optional banner file containing a message to be

# sent to clients before they connect, such as "/etc/issue.net"

DROPBEAR\_BANNER=""

# RSA hostkey file (default: /etc/dropbear/dropbear\_rsa\_host\_key)

#DROPBEAR\_RSAKEY="/etc/dropbear/dropbear\_rsa\_host\_key"

# DSS hostkey file (default: /etc/dropbear/dropbear\_dss\_host\_key)

#DROPBEAR\_DSSKEY="/etc/dropbear/dropbear\_dss\_host\_key"

The first thing we can configure in this file is the DROPBEAR\_PORT variable, which is used to set the port the daemon should be listen to (default is port 22).

The DROPBEAR\_EXTRA\_ARGS variable can be used to specify the options that will be passed to dropbear. Say for example we want to disable password login. We can accomplish the task by using the -s option (consult the dropbear manpage for a complete options list), therefore we write:

DROPBEAR\_EXTRA\_ARGS="-s"

The DROPBEAR\_BANNER option can be used to specify a file containing a message to be displayed to clients when they try to connect to the server (the same can be done by using the -b option).

Finally, with the DROPBEAR\_RSAKEY and DROPBEAR\_DSSKEY variables, we can specify alternative paths for the RSA and DSS server keys, the default being /etc/dropbear/dropbear\_rsa\_host\_key and /etc/dropbear/dropbear\_dss\_host\_key respectively. The keys are automatically generated during the program installation by the dropbearkey utility (keep reading to learn how to use it).

On Fedora, the options are managed in a different way. If we take a look at the dropbear systemd unit used to configure the service we can observe the following directives:

$ systemctl cat dropbear.service

systemctl cat dropbear

# /usr/lib/systemd/system/dropbear.service

[Unit]

Description=Dropbear SSH Server Daemon

Documentation=man:dropbear(8)

Wants=dropbear-keygen.service

After=network.target

[Service]

EnvironmentFile=-/etc/sysconfig/dropbear

ExecStart=/usr/sbin/dropbear -E -F $OPTIONS

[Install]

WantedBy=multi-user.target

If we look at the [Service] stanza, we can see the EnvironmentFile directive which is used to specify a file sourced for environment variables. In this case the file is /etc/sysconfig/dropbear (it doesn't exist by default, therefore we must create it). As we can deduce observing the ExecStart instruction, the command options are passed via the expansion of  the $OPTIONS variable: it must be defined inside the file mentioned above.

Let see an example. Suppose we want to display a message when an user tries to connect. To accomplish the task we must use the dropbear -b option and specify a file containing the message to be displayed as an argument. Assuming this file is "/etc/banner" (the path is arbitrary), inside the /etc/sysconfig/dropbear file we write:

OPTIONS="-b /etc/banner"

Each time we make a change, we must restart the service to make it effective. We will see how to do it in the next paragraph.

**Manage the dropbear server**

On some distributions, like Ubuntu, the dropbear daemon is automatically started and enabled on boot automatically during the installation. To verify the state of the dropbear service, we can run the following commands:

# Check if the service is active

$ systemctl is-active dropbear

active

# Check if the service is enabled

$ systemctl is-enabled dropbear

enabled

To activate or enable the service manually we use the following commands:

# Start the service

$ sudo systemctl start dropbear

# Enable the service at boot

$ sudo systemctl enable dropbear

# Do both actions with one command:

$ sudo systemctl enable --now dropbear

As already said, whenever we change a configuration parameter, we need to restart the server. All we have to do is to run:

$ sudo systemctl restart dropbear

**Dropbear utilities**

The dropbear application comes with some useful utilities. Let's take a look:

**dropbearkey**

We already saw dropbear-key is used to generate private server keys. When using the utility we must specify the type of key to generate, one among rsa, ecdsa and dss with the -t option and a destination file to be used for the secret key. We can also specify the key size in bits (it should be a multiple of 8), using the -s option. Let's see an example.

To generate a 4096 bits private rsa key to a file named "key" we can run:

$ dropbearkey -t rsa -s 4096 -f key

The command generates the key and displays its public portion onscreen. This part of the key can be also visualized it later, using the -y option of dropbearkey. The option can be useful, for example, to generate a file containing the public key. All we have to do is to redirect the output of the command. We can run:

$ dropbearkey -y -f key | grep ^ssh-rsa > key\_public

**dropbearconvert**

The dropbearconvert utility is used to convert between Dropbear and OpenSSH private keys formats. When using the application we need to provide:

* input\_type: the type of the key that should be converted, it can be either dropbear or openssh;
* output\_type: the type the key should be converted to, either dropbear or openssh;
* input\_file: The path of the key to be converted;
* output\_file: The destination path for the converted key.

**dbclient**

To connect to a dropbear ssh server, we can use both ssh, which is the client provided by OpenSSH, or the native dropbear client: dbclient. The latter supports all the options we would expect. Among the others, we can use the  -p option to specify an alternative server port to connect to, or -i to specify an identity file to use for the connection. To connect to a dropbear server using dbclient we can run:

$ dbclient egdoc@192.168.122.176

Host '192.168.122.176' is not in the trusted hosts file.

(ecdsa-sha2-nistp521 fingerprint md5

5e:fa:14:52:af:ba:19:6e:2c:12:75:65:10:8a:1b:54)

Do you want to continue connecting? (y/n) y

egdoc@192.168.122.176's password:

**Conclusion**

In this tutorial we learned to know dropbear, a lighter alternative to the [openssh](https://linuxconfig.org/most-common-custom-ssh-configurations-of-the-openssh-server) server. Dropbear comes with a complete set of features, like X11 forwarding, and is especially fit to be installed on systems with limited resources, like routers or embed devices. We saw how to install the program on the major Linux distributions, how we can modify the server behavior by specifying the options it should be run with.  
  
Finally we took a look at some utilities that come with the dropbear suite, such  as dropbearkey, dropbearconvert and dbclient. The first two are used to generate private keys and to convert a key from the openssh format to the dropbear format (or vice versa), respectively. The third is a small client which can be used as an alternative to ssh.