

EDMdesigner **EDMdesigner** [Menu](#)
[Log in/Sign Up](#)

- [API](#)
- [APP](#)
- [ECMS](#)
- [JOBS](#)
- [BLOG](#)

16 Command Examples to Send Email From The Linux Command Line

[Email sending](#), [SMTP](#), [Linux](#), [Command Line](#), [Terminal](#), [CLI](#)

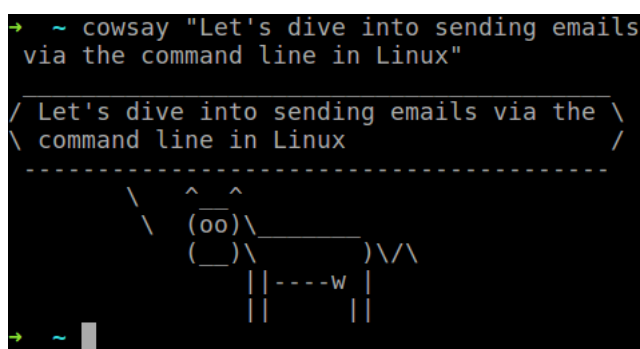
11 April 2018

In this post you'll learn how to send emails from the Linux command line. I'll show the most often used commands, SMTP configuration and terminal options.

Share and Enjoy

Whether you are a developer working non-stop in the Linux command line or a server administrator wanting logs or other data extracted from administered systems, knowing how to send email from the command line is greatly useful.

In this post, you'll find examples of how to send email from the terminal. We'll look at custom configurations and touch on how to set up SMTP connections to email accounts or email service providers. I'll show you how to install the necessary console mailer packages and provide background for command line options based on the package's manuals.



Setting Up Command Line Mailer Packages

All the mailer packages throughout the tutorial are available through [Debian's Advanced Package Manager](#) so the installation steps in the example below may be used for all the showcased packages.

Please note that these pieces of software may be installed by other default package managers such as `yum` or `dnf` depending on your Linux system distribution.

In most cases, either `sendmail` or `postfix` is available in each Linux distribution out-of-the-box or at least that's what similar blog posts say. Either I'm unlucky or the others were wrong, but I had to install one manually. **However, you definitely need one of them for email delivery.**

Setting Up Sendmail with SSMTP Package

I'll continue with `sendmail` and the SSMTP package. SSMTP contains the `sendmail` package under the hood as you'll see and it facilitates the configuration, which is pretty hard for `sendmail` otherwise.

Step 1. Check if Command Line Mailer Package is Installed

Type in the mailer package's name just the way you would run the command, to check if it's available:

- If the package is installed, the command runs, and the prompt changes — while trying to execute the command — and you'll see a blinking cursor followed by the message Recipient names must be specified in the case of `sendmail`. Conclusion: the package is installed.
- If the package is NOT installed, you'll receive a warning message and a suggestion how to install it:

```
$ sendmail
```

The program 'sendmail' can be found in the following packages:

```

* exim4-daemon-heavy
* exim4-daemon-light
* postfix
* citadel-mta
* courier-mta
* dma
* esmtp-run
* masqmail
* msmtm-mta
* nullmailer
* opensmtpd
* qmail-run
* sendmail-bin
* ssmtp
  
```

Try: `sudo apt install <selected package>`

In this example, I tried if the `sendmail` command works. After we add the SSMTP package, this command and the `ssmtp` command will both work and change the command prompt in the terminal. Let's look at that!

Step 2. Installing Sendmail Command Line Mailer Package

Send us your message now!

As the terminal output suggested, we need to run the Advanced Package Manager's install command — usually with root permission — to install the necessary package. As mentioned before, the configurations are much simpler for sendmail when you abstract away the complexity of the configuration. That can be done with SSMTP:

```
sudo apt install ssmtp
```

Step 3. Verify the Installation

After the installation completed, you can check if sendmail is able to forward messages. Type the below code in the terminal:

```
echo "Subject: hello" | sendmail test@example.com
```

It should send out the email if everything works correctly. Here the "hello" string piped to the sendmail command will be the subject of the sent message, while the defined email address is naturally the recipient of the email.

You run the command... and, oops: sendmail: Cannot open mailhub:25. The reason for this is that we didn't provide mailhub settings at all. In order to forward messages, you need an SMTP server configured. That's where SSMTP performs really well: you just need to edit its configuration file once, and you are good to go.

Step 4. Configuring SSMTP

Let's locate the config file at /etc/ssmtp/ssmtp.conf. Here, you should put the code below to configure an SMTP relay:

```
UseSTARTTLS=YES
root=<your-email-address>
mailhub=smtp.gmail.com:587
AuthUser=<your-account's-user-name>
AuthPass=<your-account's-password>
```

This means that you set up an external service that will actually forward your emails. When you provide credentials to Gmail for example, you'll see the messages sent from the terminal in your mailbox's 'Sent mail' directory. [The configurations look similar](#) with other services too.

This was the basic configuration for sendmail and the installation steps for any frequently used command line mailer package. Next, we'll jump on installing and trying out various other packages.

Wanna speed up your email production process?

Produce HTML emails faster with the collaborative email builder and content management platform freshly created by EDMdesigner team

Try Chamaileon for free

Sending Email Using the SSMTP Command

This is a nice little tool we just configured for mail delivery. [As its man page suggests](#), it's a minimalistic emulator of sendmail. As such, SSMTP allows users to transfer emails through an SMTP server from the Linux command line. It provides the means to connect to a mailhub with a proper configuration file. If your config file was set up right, all your worries regarding command line email sending can disappear.

Basic Implementation of the SSMTP command:

If you just use SSMTP, the command should look as follows:

```
ssmtp test@example.com < mail.txt
```

Following the ssmtp command, you should put the recipient address, then you can include a file that will show in the email body. Here you can define headers for the message, list further recipients and set the content type. This way you can send HTML emails. SSMTP will interpret your message and use the provided values properly. Here is an example of an HTML you might send:

```
Cc: example@edmdesigner.com
Subject: This is an HTML email
From: developer@example.com
Content-Type: text/html; charset="utf8"

html>
<body>
<div style="
    background-color:
    #abcdef; width: 300px;
    height: 300px;
">

</div>
You can add any valid email HTML here.
</body>
</html>
```

You can't really give further options with SSMTP as they are not respected by its minimalistic design.

Troubleshooting:

In case you run into the following error on the command line:

```
ssmtp: Authorization failed (534 5.7.9
http://support.google.com/accounts/bin/answer.py?answer=185833
bk8sm8525341pad.28 - gsmtp)
```

that's likely because Gmail is not treating SSMTP as a secure application. To get rid of this error, [you need to change the settings](#) in your Gmail account – you can look up more information on potential risks there as well.

Send us your message now!

Sending Email Using the Sendmail Command

For quite some time now sendmail is the classical [mail transfer agent](#) from the world of UNIX. It was first introduced in 1979 and its highly-configurable nature and scalability made it the default go-to for server administrators.

We've already seen how to send basic emails by the `ssmtp` command, so let's see the difference using `sendmail`:

```
echo "Subject: hello" | sendmail -v test@example.com < mail.txt
```

I threw in the `-v` argument, which will make the communication between the mail server and your mail transfer agent visible.

There are plenty of configurations that you can use if you set up your own mail server and implement sendmail on it. However, that is out of the scope of this article. We will return to the topic in an upcoming one. Stay tuned.

Create great looking emails with less effort?

Produce HTML emails easier with the email builder and content management platform packed with 100+ ready made templates

Try Chamaileon for free

Sending Email Using CURL Command

This tool is also tremendously common for data transfer from a server. It supports many protocols, like HTTP, FTP, POP3 or SMTP. The [CURL](#) package is used widely around the globe, one main reason is that it has [native PHP implementation](#) and [PHP was the default server-side scripting language](#) for a long time.

Installation:

Check if the package is already installed. If not run the below:

```
sudo apt install curl
```

When you're done, you can access the cheat sheet for the available options for CURL by running `curl -h`. The more detailed version is accessible by either `man curl` or `curl --manual`. If you want to gain in-depth knowledge using cURL, there's [this handy Ebook](#) that contains everything you would ever want to know.

Basic implementation for email sending with CURL:

```
curl --url 'smtps://smtp.gmail.com:465' --ssl-reqd \
--mail-from 'developer@gmail.com' --mail-rcpt 'edm-user@niceperson.com' \
--upload-file mail.txt --user 'developer@gmail.com:your-account-password'
```

In order to send an email with CURL, you need to set up SMTP connection. Most often [Google's](#) or [Yahoo's](#) outgoing mail servers are used for testing email sending with SMTP. Please note that you must turn on access for less secure apps in Gmail settings and similar additional security settings may apply for Yahoo as well.

In the terminal command snippet above, the `--url` and `--user` parameters define the SMTP connection settings. The password section for `--user` parameter is your account's password for the given email address. Naturally, you could also use cloud email service providers. I will show that using Mailgun in the following section.

Advanced implementation for email sending with CURL:

Working with email delivery platforms, you first need to obtain an API key. In this tutorial, I'll show CURL email sending example using Mailgun, so if you get stuck during the registration process [follow this setup guide](#) to get on the right track.

```
curl -sv --user 'api:key-7e55d003b...f79accd31a' \
https://api.mailgun.net/v3/sandbox21a78f824...3eb160ebc79.mailgun.org/messages \
-F from='Excited User <developer@yourcompany.com>' \
-F to=sandbox21a78f824...3eb160ebc79.mailgun.org \
-F to=user@gmail.com \
-F subject='Hello' \
-F text='Testing some Mailgun awesomeness!' \
--form-string html='<h1>EDMdesigner Blog</h1><br /><cite>This tutorial helps me understand email sending from Linux console</cite>' \
-F attachment=@logo2yellow.jpg
```

The syntax is pretty straightforward. You may have noticed before, but you need to end each line of the command with `\` characters. This is a line continuation character, the command can be run without them all in one line as well. The other thing to remember is the `-F` option. An excerpt from the man page:

```
-F, --form <name=content>
(HTTP) This lets curl emulate a filled-in form
in which a user has pressed the submit button.
This causes curl to POST data using the Content-Type
multipart/form-data according to RFC 2388.
This enables uploading of binary files etc.
To force the 'content' part to be a file, prefix
the file name with an @sign.
```

As I encouraged before, it's a good practice to dive into the manuals when using these command line mailer commands. As the CURL manual says, basically you're creating a form to send. Each line is an entry given as a `name=content` pair. When you attach files from your local machine, you need an `@` in front of the file name.

This is how the delivered email will look like:

Hello Inbox x



Excited User via mailgun.org

to me ▾

EDMdesigner Blog

This tutorial helps me understand email sending from Linux console



You could do the same using most of the other email delivery platforms, like [Sendgrid](https://sendgrid.com) for example.

Sending HTML email from the command line with CURL:

You can apply the same logic of building a form in a text file, which enables you to send HTML messages easily:

```
To: customer@gmail.com
Subject: This is an HTML message
From: developer@yourcompany.com
Content-Type: text/html; charset="utf8"

<html>
<body>
<div style="
    background-color:
    #abcdef; width: 300px;
    height: 300px;
">

</div>
You can add any valid email HTML here.
</body>
</html>
```

If you run the simple command from the first example...

```
curl --url 'smtps://smtp.gmail.com:465' --ssl-reqd \
--mail-from 'developer@edmdesigner.com' --mail-rcpt 'edm-user@niceperson.com' \
--upload-file mail.txt --user 'developer@gmail.com:your-account-password'
```

... and direct the `--upload-file` parameter to the updated text file (mail.txt in our case), the received message will show the HTML content.

Troubleshooting:

You may need the `--insecure` switch, which allows CURL to perform "insecure" SSL connections and transfers. CURL will return an error message if the remote server is using a self-signed certificate, or if the remote server certificate is not signed by a CA listed in the CA cert file.

Sending Email Using Swaks command

As their documentation states, Swaks is a flexible, scriptable, transaction-oriented SMTP test tool. It is able to handle SMTP features and extensions such as TLS, authentication, and pipelining and multiple version of the SMTP protocols. It also supports multiple transport methods including UNIX-domain sockets, internet-domain sockets, and pipes to spawned processes.

Installation:

```
sudo apt install swaks
```

Basic implementation for email sending with Swaks:

In order to connect to an SMTP account you need to provide the server `-s`, the user `-au`, the password `-ap` and the address `-t` (where you want to send your mail) flags. The [-tls flag is also important](#) if you connect on port 587.

```
swaks --to mailbox@example.com -s smtp.gmail.com:587
-tls -au <user-account> -ap <account-password>
```

The credentials can also be provided via command line prompts, if you only specify to authenticate `-a`, not filling in the user and password through command line options.

Advanced implementation for email sending with Swaks:

Send us your message now!

```
swaks --to mailbox@example.com -s smtp.gmail.com:587 -tls -au <user-account> -ap <account-password> --attach -d ./mail.txt
```

There are other options which we will not touch now. Let's head to our next mailer package!

Sending Email Using the Mutt Command

[Mutt](#) is different from the previous email clients, as it is a feature-rich command line email client. You can use it for reading emails from the Linux terminal, connecting to local user mailboxes or to POP/IMAP servers. Mutt supports several mailbox formats such as mbox, MH, maildir, MMDF, full control of message headers during email composition, and multiple message tagging and colors to handle messages.

Installation:

```
sudo apt install mutt
```

Basic implementation for email sending with mutt:

By this time you won't be surprised by the structure of the command. You can provide an empty message body with `< /dev/null`:

```
# mutt -s "Test Email" user@example.com < /dev/null
```

Advanced implementation for email sending with mutt:

For sending email including attachments, you need the `-a` flag. This way you can send useful files, like system logs, to a specified address.

```
# mutt -s "System logs" -a /opt/backup.sql user@example.com < /dev/null
```

You could also send HTML email if the file extension you add to the message body is in HTML format. Mutt will recognize the file type and resolve it so it won't send your HTML as plain text:

```
mutt -s "Email subject" test@example.com < email.html
```

This is it for email sending with mutt. If you have it on your machine, I'd suggest to try out how you can work with incoming configuring your mailbox and handling incoming messages. If you do so, please share your thoughts in the comment section below. I welcome any opinion about the packages shown in the tutorial.

Sending Email Using the Mailx Command

[The Mailutils](#) is a compound package for multiple use cases. It's designed to serve regular users, system administrators, and developers. We'll only touch on its basic functionalities regarding email forwarding. If you are interested in more features, check out its `man` page.

Installation:

```
sudo apt install mailutils
```

Basic implementation for email sending with mailx:

The mailx package has 2 equal command syntax (`mail` and `mailx` both work in the exact same way). The most simplistic command consists of the `mail/mailx` command and the address to send the mail to, but you would probably include the subject and at least some text-based message body. Such a command should look like this:

```
echo "message body" | mail -s "subject" test@example.com
```

By using the `echo` command and piping the output to the `mail` command, you can avoid mailx's additional prompts for Cc addresses and the message body.

Advanced implementation for email sending with mailx:

For the advanced example, we will send a full-fledged HTML email again. We need to provide the email HTML file of our choice with the `<` character and add the "append flag" with the content type. The full code looks as follows:

```
mailx -a 'Content-Type: text/html'
-s "This is advanced mailx indeed!" < email.html
"recipient1@gmail.com, recipient2@yahoo.com"
```

You can also set a return address with `--return-address=<EMAIL-ADDRESS>`, which may be useful. Another useful option, which I couldn't resolve to work with HTML email parallel, is to provide attachments in the same message: `-A "mail.txt"`. Though you can list multiple recipients, only a single attachment can be attached. You could configure external SMTP server if that fancies you.

Using Telnet Alternative Openssl Command

[Telnet is an interactive communication protocol](#) for communication with another host. It could be used to open an SMTP connection to another server and transfer email messages. However, as the protocol is not secured, most servers will reject this communication type. This is where [Openssl](#) comes into the picture as it's built with [SSL/TLS](#) security included. It is invoked similarly to telnet with a host argument, and it performs an open command implicitly:

```
smiska@Desktop(emails) openssl s_client -connect smtp.gmail.com:465
--
-crlf -ign_eof
---
certificate negotiation output from openssl
---

220 smtp.gmail.com ESMTP j92sm925556edd.81 - gsmt
EHLO localhost
250-smtp.gmail.com at your service, [78.139.22.28]
250-SIZE 35882577
250-8BITIME
250-AUTH LOGIN PLAIN XOAUTH2 PLAIN-CLIENTTOKEN OAUTHBEARER XOAUTH
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-CHUNKING
250 SMTPUTF8
AUTH PLAIN *passwordhash*
235 2.7.0 Accepted
MAIL FROM: <developer@edmdesigner.com>
250 2.1.0 OK j92sm925556edd.81 - gsmt
```

Send us your message now!

```
rcpt to: <friendly@user.com>
250 2.1.5 OK j92sm925556edd.81 - smtp
DATA
354 Go ahead j92sm925556edd.81 - smtp
Subject: This is openssl mailing

Hello nice user
.
250 2.0.0 OK 1339757532 m46sm11546481eeh.9
quit
221 2.0.0 closing connection m46sm11546481eeh.9
read:errno=0
```

I indented the inputs you need to provide while the communication channel is open. These inputs are self-explanatory, but one thing I need to detail is the password hash. It can be a base64 encoded hash, which you can obtain with the following command: `echo -en '\000username@gmail.com\000gmailpassword' | base64`. The escape characters from the string can't be omitted. If you don't have the base64 package installed, it is available through package managers.

As you can see above, this means of email sending only provides an interface for text-based messages, therefore it has its limitation in use cases. On the other hand, it shows all information exchanged with the mail server, which makes debugging really easy.

Troubleshooting Mail Problems

Check the mail logs

You need to locate your log file by navigating to the corresponding folder. I prefer to use [my code editor \(eg. Sublime Text\)](#) for this so I can have the logs with syntax highlight:

```
$ sudo subl /var/log
```

Depending on your current Linux distribution you [may need to look at different source](#).

The log may contain useful information about deliverability problems. At this point, when you scroll to the bottom of this:

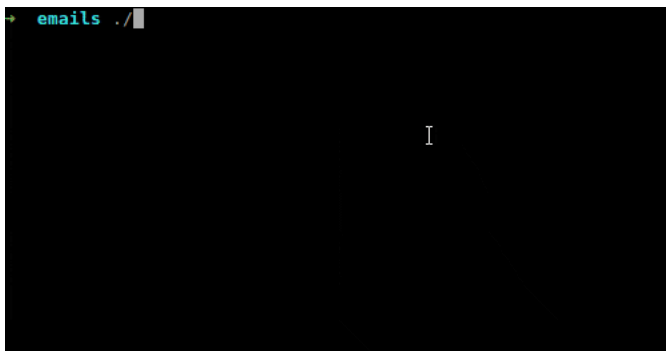
```
952 Feb 20 17:15:26 smiska-K401LB sm-mta[17488]: w1J7AoQG013569: 5: fl=0x8000, mode=100640: dev=8/5, ino=14168236, nlink=1
953 Feb 20 17:15:26 smiska-K401LB sm-mta[17488]: w1J7AoQG013569: 6: fl=0x8000, mode=100640: dev=8/5, ino=27266224, nlink=1
954 Feb 20 17:15:26 smiska-K401LB sm-mta[17488]: w1J7AoQG013569: 7: fl=0x8000, mode=100640: dev=8/5, ino=27266224, nlink=1
955 Feb 20 17:15:26 smiska-K401LB sm-mta[17488]: w1J7AoQG013569: 9: fl=0x1, mode=10600: FIFO: dev=0/10, ino=114197, nlink=
956 Feb 20 17:15:26 smiska-K401LB sm-mta[17488]: w1J7AoQG013569: 10: fl=0x0, mode=10600: FIFO: dev=0/10, ino=114198, nlink=
957 Feb 20 17:15:26 smiska-K401LB sm-mta[17488]: w1J7AoQG013569: MCI@0x0: NULL
958 Feb 20 17:15:26 smiska-K401LB sm-mta[17488]: w1J7AoQG013569: MCI@0x0: NULL
959 Feb 20 17:15:26 smiska-K401LB sm-mta[17488]: w1J7AoQG013569: to=<smiska@smiska-K401LB>, delay=1+08:53:59, xdelay=00:00:0
960 Feb 20 17:15:26 smiska-K401LB sm-mta[18825]: w1J70oct002561: Warning: program /usr/sbin/sensible-mda unsafe: No such fil
961 Feb 20 17:15:26 smiska-K401LB sm-mta[18825]: w1J70oct002561: SYSERR(root): Cannot exec /usr/sbin/sensible-mda: No such f
962 Feb 20 17:15:26 smiska-K401LB sm-mta[17488]: w1J70oct002561: SYSERR(root): putbody: write error: Broken pipe
963 Feb 20 17:15:26 smiska-K401LB sm-mta[17488]: w1J70oct002561: 0: fl=0x8000, mode=20666: CHR: dev=0/6, ino=6, nlink=1, u
964 Feb 20 17:15:26 smiska-K401LB sm-mta[17488]: w1J70oct002561: 1: fl=0x8001, mode=20666: CHR: dev=0/6, ino=6, nlink=1, u
965 Feb 20 17:15:26 smiska-K401LB sm-mta[17488]: w1J70oct002561: 2: fl=0x8001, mode=20666: CHR: dev=0/6, ino=6, nlink=1, u
966 Feb 20 17:15:26 smiska-K401LB sm-mta[17488]: w1J70oct002561: 3: fl=0x2, mode=140777: SOCK localhost->[[UNIX: /run/syst
967 Feb 20 17:15:26 smiska-K401LB sm-mta[17488]: w1J70oct002561: 4: fl=0x8002, mode=100640: dev=8/5, ino=14163935, nlink=1
```

you can inspect if delivery is successful or understand why the process failed.


It can happen that your console shows no errors, but in the mail log you can see that the message was queued for delivery, so the process didn't halt on your machine but probably was denied by the mail server you were trying to reach.

Running All Command in a Sequence

So, everything is ready for the final test. All the mailer terminal commands are gathered here in a script file. Once you downloaded [the working folder](#) with the script file in it, navigate to the containing folder and make the script executable: `chmod u+x mail.sh`. If you have all the files I provided ready, you can start testing the mailer commands shown in the tutorial. You just need to replace the "<RECIPIENT ADDRESS>" to your test email address in the code and run `./mail.sh` in the command line.



Once the script finishes running, your inbox will be filled with the test emails.

<input type="checkbox"/> ☆ smiska	› 15		10:52 pm
<input type="checkbox"/> ☆ smiska	› 14		10:52 pm
<input type="checkbox"/> ☆ smiska	» 13		10:52 pm
<input type="checkbox"/> ☆ smiska	» 12		10:52 pm
<input type="checkbox"/> ★ smiska (2)	» 11		10:52 pm
<input type="checkbox"/> ★ smiska	» 9		10:52 pm
<input type="checkbox"/> ☆ smiska	» 8		10:52 pm
<input type="checkbox"/> ☆ me	› 7		10:52 pm

Well done!

Summary

In this article, I showed you many ways to send email from the Linux command line. Hopefully, based on what you saw, you can install and configure these services on your machine yourself as well.

The post showed the most commonly used command line mailer packages and how to send emails with them, and in the end, I created a shell script that gathers all the commands used in the article. You can download [the projects working files here](#), and if you install the packages introduced in the tutorial and also fill in your credentials and recipient details, you can run the script at your leisure.

Could you follow each step of the tutorial? Very well then, it makes me glad. In case you have questions or see an opportunity to improve the content further by your ideas, please head to the comment section below.

Thanks for your attention. Happy sending!

Author



[Mihály Sáróy](#)

Developer @ EDMdesigner.com

Send us your message now!

Comments

Community

Login

Recommend 2

Tweet

Share

Sort by Best

Join the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS

Name

**Rick Lee** • a year ago

Shouldn't "html>" really be "<html>"?

9 ^ | v • Reply • Share ›

**Ran Talbott** • 10 months ago

I tried installing SSMTP, and it created a mess: it removed the original sendmail, and now "simplified" version it installed is trying to send mail to local mailboxes via gmail. Which, of course, bounces them because names like "localhost"

and my Linux system's hostname fail global DNS lookups.

This is not a good option if you're running a Linux system.

^ | v • Reply • Share ›

**Capados** • a year ago

Hi !

The mailx command does not work :

" mailx -a 'Content-Type: text/html'

-s "This is advanced mailx indeed!" < email.html

"recipient1@gmail.com, recipient2@yahoo.com""

the -a option is for attaching file. Where did you find this ? I see many wrong recommendation on the internet based on the -a option

^ | v • Reply • Share ›

**Tina Head** • a year ago

Thank you for this tutorial, saved me a lot of time

Personally, I also use <https://smtpwarmup.com> tool to warm up IPs faster

^ | v • Reply • Share ›

**Jord Wegge** • 2 years ago

Very nice article; real good, thanks for the detailed overview...

I found one tiny error though, you need a double hyphen between attachment path & adress for mutt.

mutt -s "System logs" -a /opt/backup.sql

user@example.com < /dev/null

-->

mutt -s "System logs" -a /opt/backup.sql --

user@example.com < /dev/null


Share and Enjoy

- [16 COMMAND EXAMPLES TO SEND EMAIL FROM THE LINUX COMMAND LINE](#)
 - [Setting Up Command Line Mailer Packages](#)
 - [Setting Up Sendmail with SSMTP Package](#)
 - [Sending Email Using the SSMTP Command](#)
 - [Sending Email Using the Sendmail Command](#)
 - [Sending Email Using CURL Command](#)
 - [Sending Email Using Swaks command](#)
 - [Sending Email Using the Mutt Command](#)
 - [Sending Email Using the Mailx Command](#)
 - [Using Telnet Alternative Openssl Command](#)
 - [Troubleshooting Mail Problems](#)
 - [Running All Command in a Sequence](#)

Send us your message now!

- [Summary](#)

[< Previous](#)[Next >](#)

-  Email Marketing and Mobile Optimization

[Read more](#)

- 16 Command Examples to Send Email From The Linux Command Line

In this post you'll learn how to send emails from the Linux command line. I'll show the most often used commands, SMTP configuration and terminal options.

[Read more](#)

- How to Create Responsive Email Designs with React Native

Responsive email coding is complex. Are you native in React Native? Abstract away email HTML & CSS into components with react-html-email or mjml frameworks.

[Read more](#)

- Company
- [Jobs](#)
- [Blog](#)
- Useful links
- [API Docs](#)
- [API Tutorials](#)
- [Compatibility](#)
- [Github](#)
- Legal
- [Terms of Use](#)
- [Privacy Policy](#)

Try the New Responsive Email Template Builder by EDMdesigner

[Get Started for Free](#)

info@edmdesigner.com [edmdesigner](#)
[Facebook](#) [Twitter](#) [LinkedIn](#) [Google+](#) [RSS](#)
2020 @ copyright - EDMdesigner

Try the New Responsive Email Template Builder by EDMdesigner

[Get Started for Free](#)

We detected that cookies are disabled in your browser. At EDMdesigner, we use cookies to improve the quality of our service. For the best experience, please enable cookies. Check our [Privacy Policy](#) for more information about cookies.

We use cookies to provide you with a more personalized service. By using our site you agree to our [cookie policy](#).

Send us your message now!