



Home > Security > Steganography – Hide Files Inside Images In Linux

SECURITY ♦ COMMAND LINE UTILITIES ♦ ENCRYPTION / DECRYPTION ♦ FAQ ♦ FORENSIC TOOLS ♦ LINUX ♦ LINUX BASICS ♦ LINUX TIPS & TRICKS ♦ OPENSOURCE ♦ TIPS AND TRICKS ♦ UNIX/LINUX BEGINNERS ♦ UTILITIES

# Steganography – Hide Files Inside Images In Linux

Written by Sk | **Published:** August 15, 2019 | **Last Updated on** August 28, 2022 | 27087 views

3 comments

3 f t in y e r s

This guide gives you a brief introduction to **Steganography**, and **hide files inside images** using different methods and tools in Linux.

123

Contents

▼

- What Is Steganography?
- Hide Files Inside Images In Linux
- ▼ ◦ Method 1
  - Method 2 - using Steghide
  - Method 3 - using Outguess
  - Method 4 - using Stegosuite
- ▼ ◦ Method 5 - using Steg
  - Hide Files Inside Images With Steg

## What Is Steganography?

**Steganography** is a process of hiding a file, an image, a video, a text inside another file. As mentioned in Wikipedia, Steganography is the combination of two Greek words, ***steganos*** which means "covered, concealed, or protected", and ***graphein*** which means "writing".

The sources said that this method was first followed by **Histiaeus**, an ancient Greek king, back in 440BC. He shaved his most trusted servant's head, and marked the secret message onto his head.

After the hair had regrown, he sent that servant to one of his Vassal to convey the message that has some information about the upcoming attack to Greece.

Now, in the modern age, Steganography has been evolved, much improved and widely used to send and receive digital secret messages by concealing them into another files.

For more details about Steganography, refer [this link](#). Let us now see how to hide files inside images in Linux.

### Disclaimer:

Steganography is a vast topic. This tutorial only covers how to hide files inside images, which is the core concept of Steganography. Please do not assume that this can't be broken by any security experts. The method described here is very basic, so even an intermediate security professional can easily break it in couple hours. This steps described below are purely for educational purpose. We are not responsible for any kind of misuse.

## Hide Files Inside Images In Linux

We can hide files inside images in different methods. Here I have given 5 methods.

### Method 1

I have one image file called **image.jpg** and a directory called **sk**. Inside this directory, there is a file called **secret.txt** which has some confidential message. This is the file that we are going to embed in the image.jpg file. You can place any number of files you want to hide inside this directory.

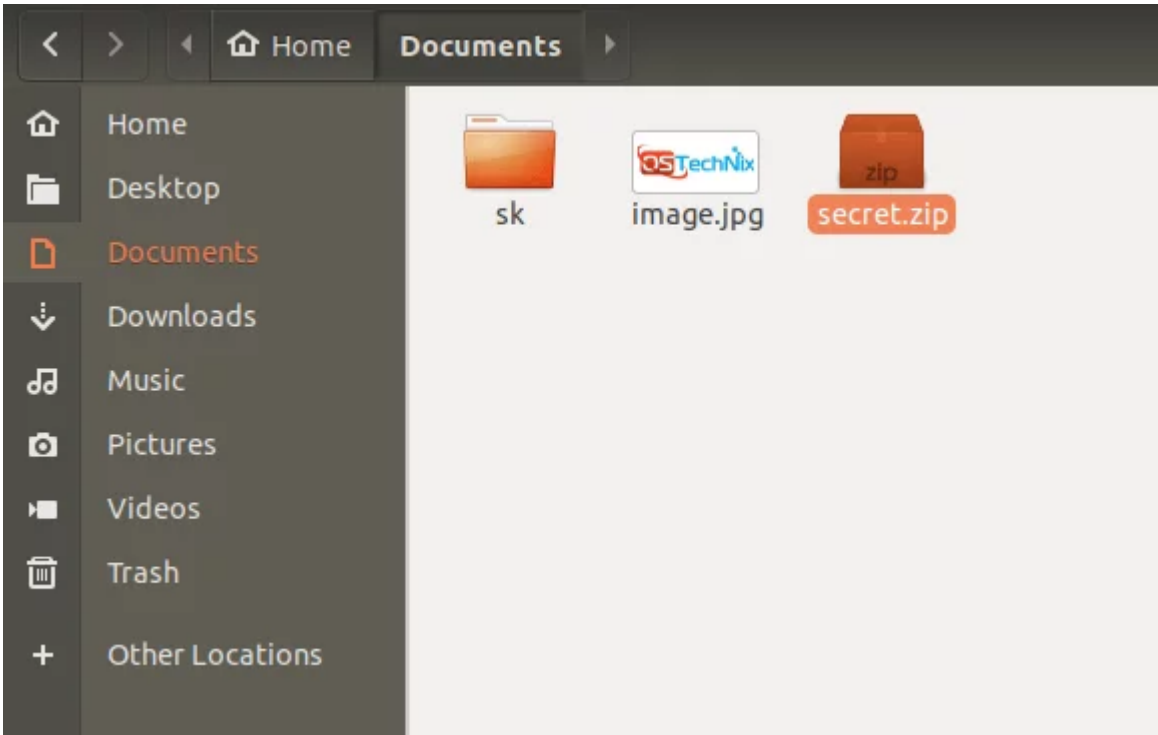
Then, I am going to compress the directory **sk** and save it as **secret.zip** to make it as single file. Finally, I will concatenate the zip file (secret.zip) and image file (image.jpg) using **cat** command and save it as **ostechnix.jpg**.

To put things more clearly,

1. **image.jpg** - A random image file.
2. **sk** - The directory that contains all secret files.
3. **secret.zip** - Archive of **sk** directory.
4. **ostechnix.jpg** - The output image file that contains both secret.zip and image.jpg.

**Step 1:** Put the image file and the directory in a folder. I have put them both in **Documents** folder.

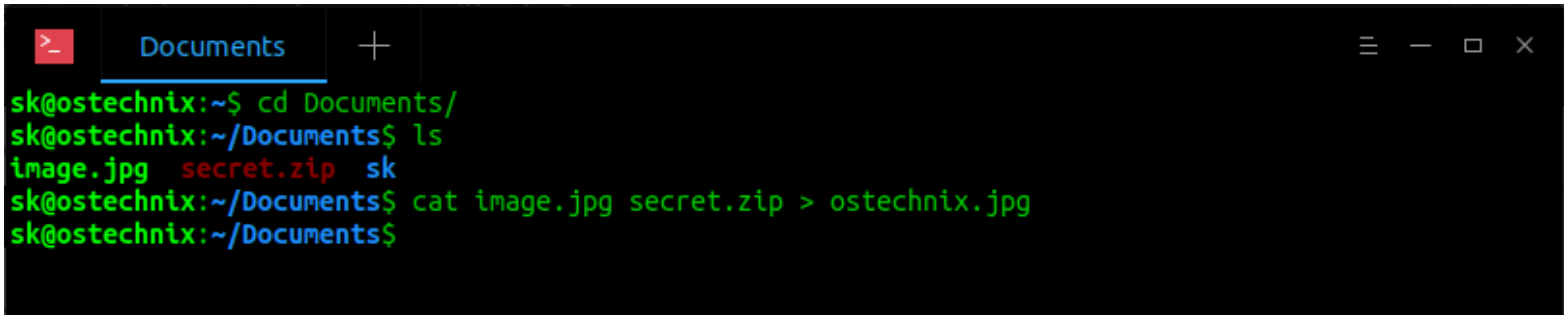
**Step 2:** Move all files you want to hide inside the folder "sk". Then, compress this folder and save it as "**secret.zip**". To compress the folder, just right click on it, and select **compress**.



**Step 3:** Next open the Terminal. Go to the location where you have stored the zip and image files (In our case it is **Documents**). Finally, concatenate the **secret.zip** and **test.jpg** files, and save them as ostechnix.jpg using **cat** command.

```
$ cd Documents
```

```
$ cat image.jpg secret.zip > ostechnix.jpg
```



Concatenate files

That's it. We have now hidden the confidential files inside **ostechnix.jpg**. It is the important file. Just delete all other files except **ostechnix.jpg**.

The **ostechnix.jpg** will look like an ordinary image file and anyone can view it using any image viewer application. But, they might not know this file has some confidential file in it.



This website uses cookies to improve your experience. By using this site, we will assume that you're OK with it. [Accept](#) [Read More](#)

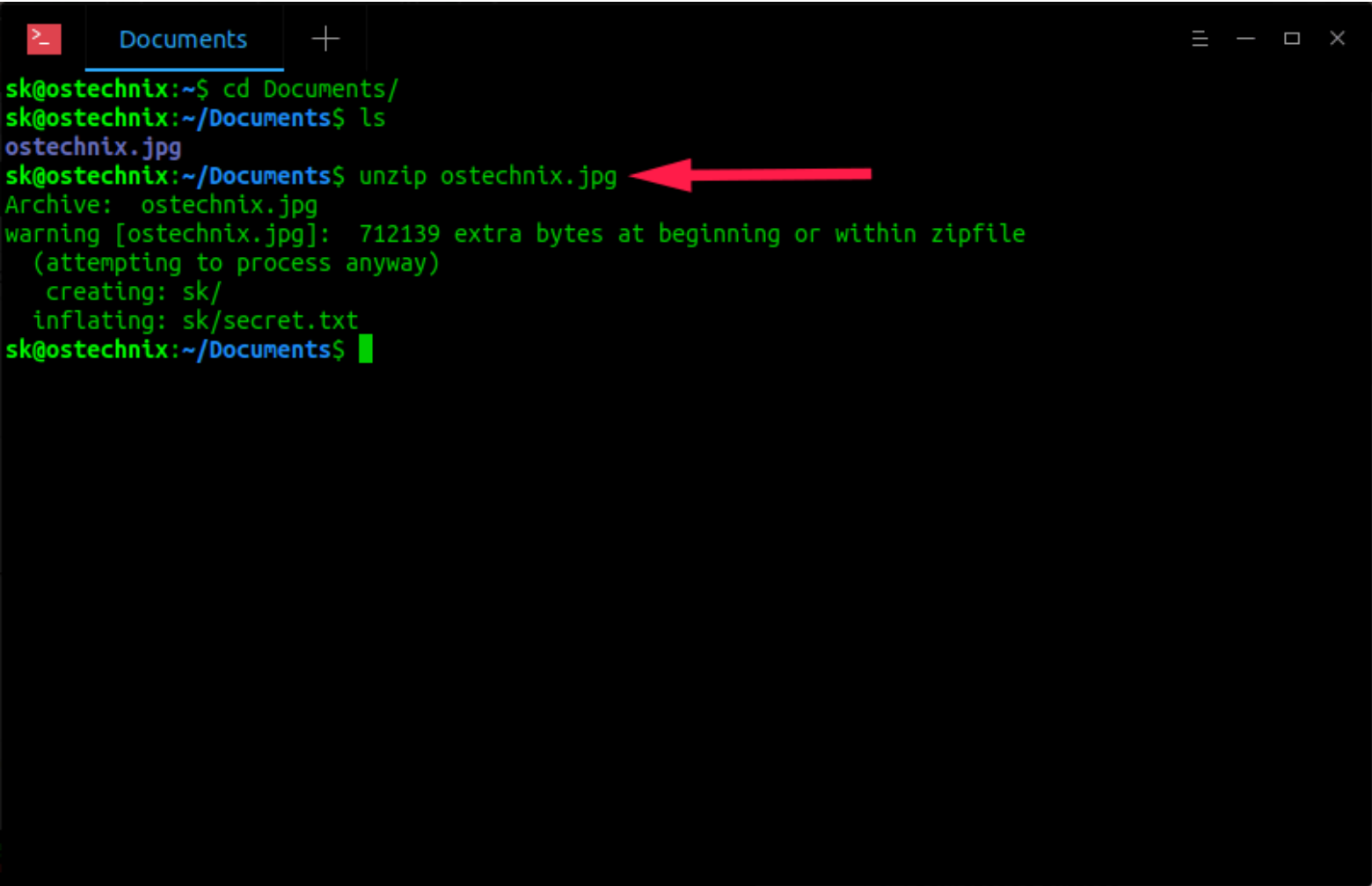
```
$ unzip ostechnix.jpg
```

Or,

```
$ unzip -t ostechnix.jpg
```

Sample output:

```
Archive:  ostechnix.jpg
warning [ostechnix.jpg]:  712139 extra bytes at beginning or within zipfile
    (attempting to process anyway)
    creating: sk/
    inflating: sk/secret.txt
```



Unzip image file

As you see in the above output, the directory **sk** that has secret files inside has been extracted. Now, go back to the folder and check the contents. You will see all the files in there.

One disadvantage of this method is we can't add any passphrase to the image file. No worries! In the following methods, we can add passphrase to the output files.

## Method 2 - using Steghide

**Steghide** is a command line utility that helps us to hide the confidential data inside an image or audio file. It supports JPEG, BMP, WAV and AU files.

Steghide is available in the default repositories of many Linux distributions.

On Arch Linux and its variants, you can install it using command:

```
$ sudo pacman -S steghide
```

On Debian, Ubuntu:

This website uses cookies to improve your experience. By using this site, we will assume that you're OK with it.

Accept

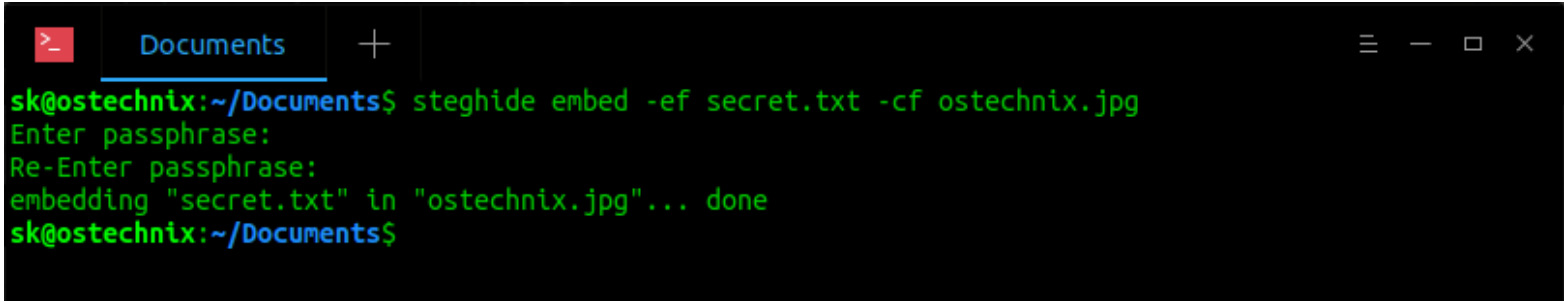
Read More

Now you can hide your confidential file inside an image or audio like below. I assume you have put the confidential file that you want to encrypt and the the image or audio file in the same folder. If you put them in the different folder, you need to give the full path in the following command.

```
$ steghide embed -ef secret.txt -cf ostechnix.jpg
```

You will be asked to enter a passphrase.

```
Enter passphrase:
Re-Enter passphrase:
embedding "secret.txt" in "ostechnix.jpg"... done
```



Hide files into image using Steghide

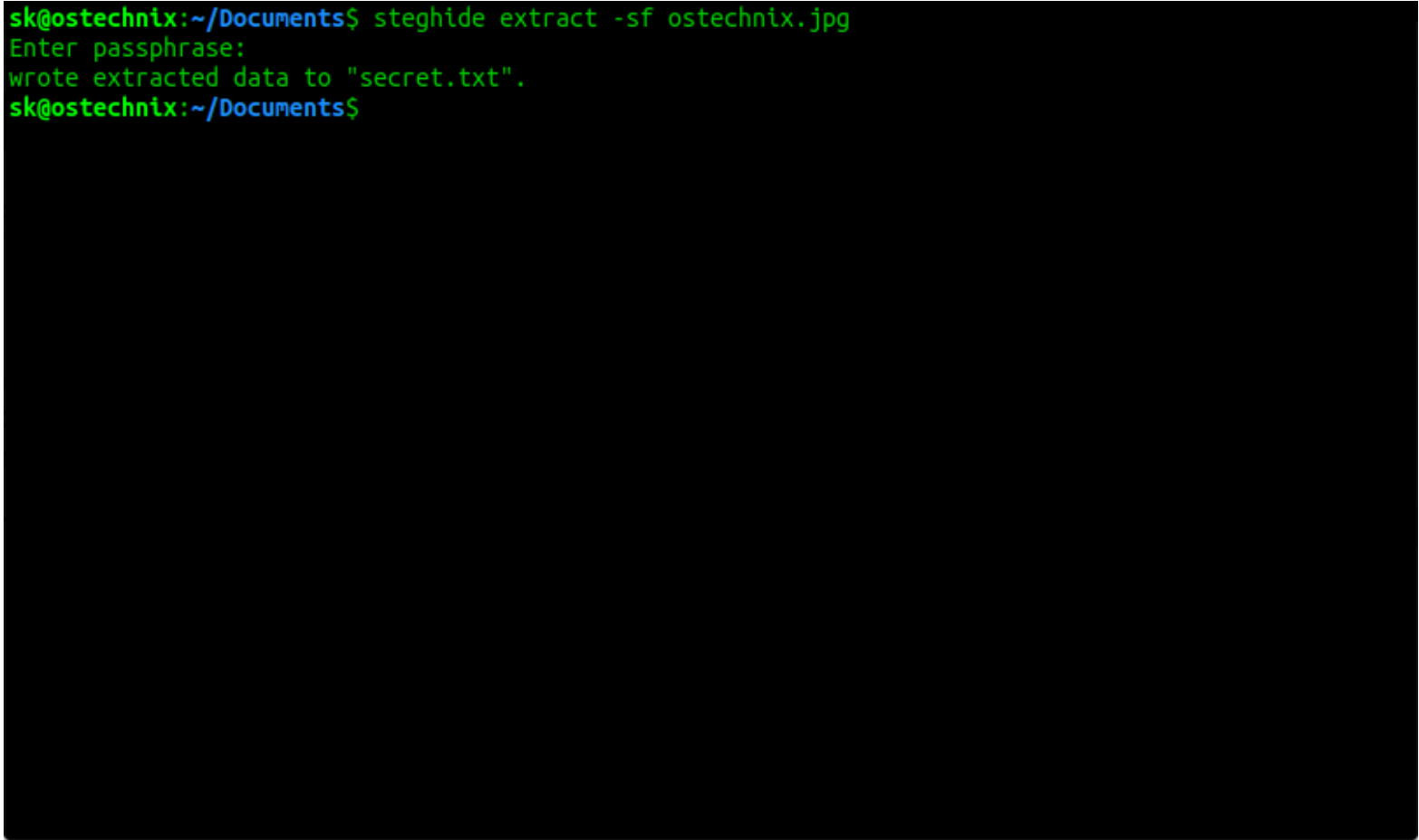
In the above example, I have embedded a text file named **secret.txt** into an image file named **ostechnix.jpg**. You can now delete the original secret.txt file. Because, we just embedded inside an image file. If you want to embed multiple files, put them in one folder and zip it and then hide it as described above.

To extract the secret file from the image, simply run:

```
$ steghide extract -sf ostechnix.jpg
```

Enter the passphrase to extract it:

```
Enter passphrase:
wrote extracted data to "secret.txt".
```



Extract files from image using steghide

For more details, refer man pages,

```
$ man steghide
```

### Method 3 - using Outguess

**Outguess** is yet another command line stegnographic tool to hide confidential files inside an image. Currently, it supports the PPM, PNM, and JPEG image formats.

To install it on Debian, Ubuntu and other DEB-based systems, run:

```
$ sudo apt install outguess
```

Once installed, go to the location where you have kept secret file and the image and embed the secret file into the image using the following command:

```
$ outguess -d secret.txt ostechnix.jpg output.jpg
```

#### Sample output:

```
Reading ostechnix.jpg....
JPEG compression quality set to 75
Extracting usable bits: 158203 bits
Correctable message size: 77641 bits, 49.08%
Encoded 'secret.txt': 160 bits, 20 bytes
Finding best embedding...
0: 88(45.8%)[55.0%], bias -17(-0.19), saved: -1, total: 0.06%
1: 90(46.9%)[56.2%], bias -27(-0.30), saved: -1, total: 0.06%
12: 85(44.3%)[53.1%], bias -36(-0.42), saved: 0, total: 0.05%
26: 91(47.4%)[56.9%], bias -45(-0.49), saved: -1, total: 0.06%
174: 87(45.8%)[54.4%], bias -48(-0.55), saved: 0, total: 0.05%
174, 39: Embedding data: 160 in 158203
Bits embedded: 190, changed: 87(45.8%)[54.4%], bias: -48, tot: 158844, skip: 15
```

```
total bits changed: 39 (change of + bias -40)
Storing bitmap into data...
Writing output.jpg....
```

Here, the **output.jpg** file is the one that has our confidential data file. keep it safe and delete everything else.

You can also add a passphrase to the output file like below.

```
$ outguess -k "my secret key" -d secret.txt ostechnix.jpg output.jpg
```

Replace "my secret key" with your own passphrase.

To extract the file, simply do:

```
$ outguess -r output.jpg secret.txt
```

Sample output:

```
Reading output.jpg....
Extracting usable bits: 158203 bits
Steg retrieve: seed: 174, len: 20
```

If you have used a passphrase, then use this command instead:

```
$ outguess -k "my secret key" -r output.jpg secret.txt
```

For more details, refer man pages.

```
$ man outguess
```

We have seen three command line utilities to hide files inside images or audio. If you don't like the command line way, here are two graphical steganographic tools.

Method 4 - using Stegosuite

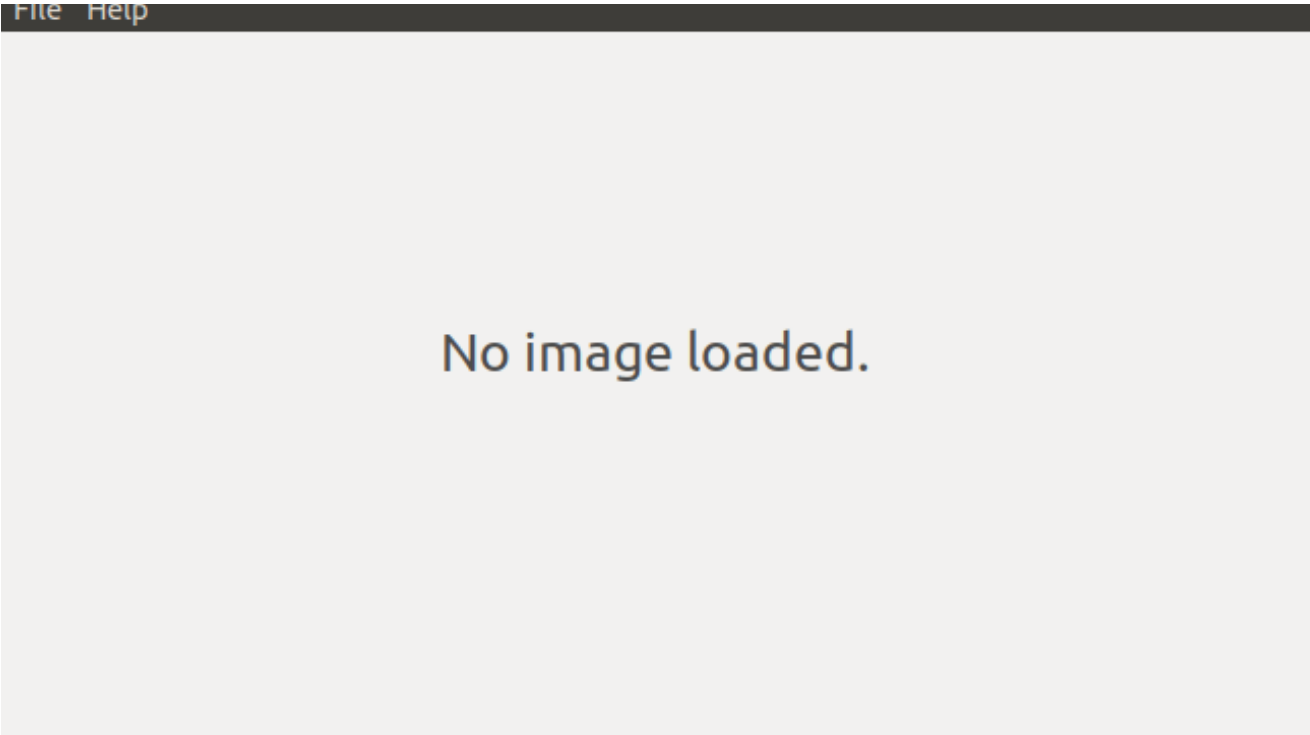
**Stegosuite** is a graphical steganographic tool to hide files inside images. It is a free and open source steganography tool written in **Java**. It uses **AES** encryption method to embed data. It supports BMP, GIF and JPG image formats.

To install it on Debian, Ubuntu and other DEB-based systems, run:

```
$ sudo apt install stegosuite
```

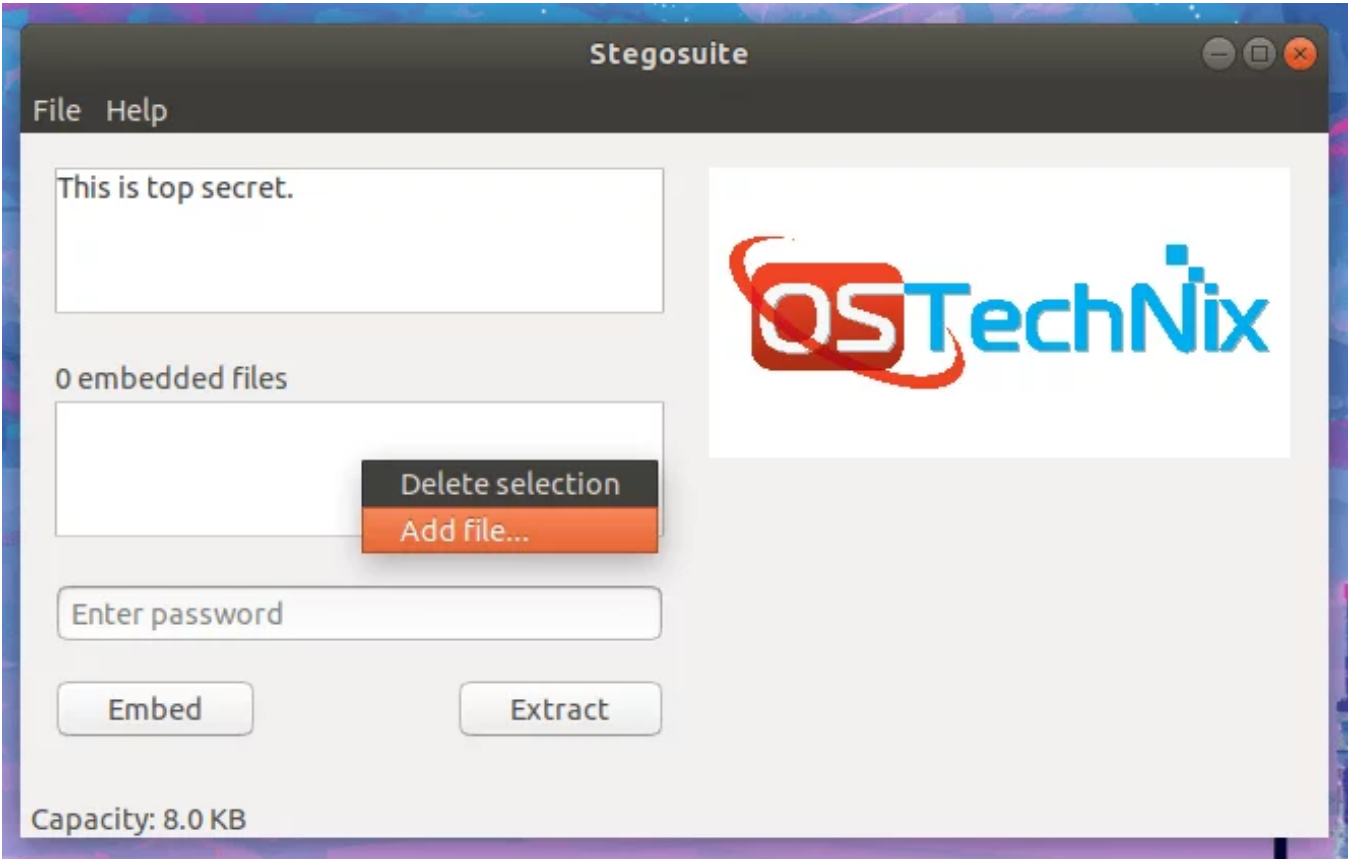
Once installed, launch Stegosuite from Dash or Menu.

This website uses cookies to improve your experience. By using this site, we will assume that you're OK with it. [Accept](#) [Read More](#)



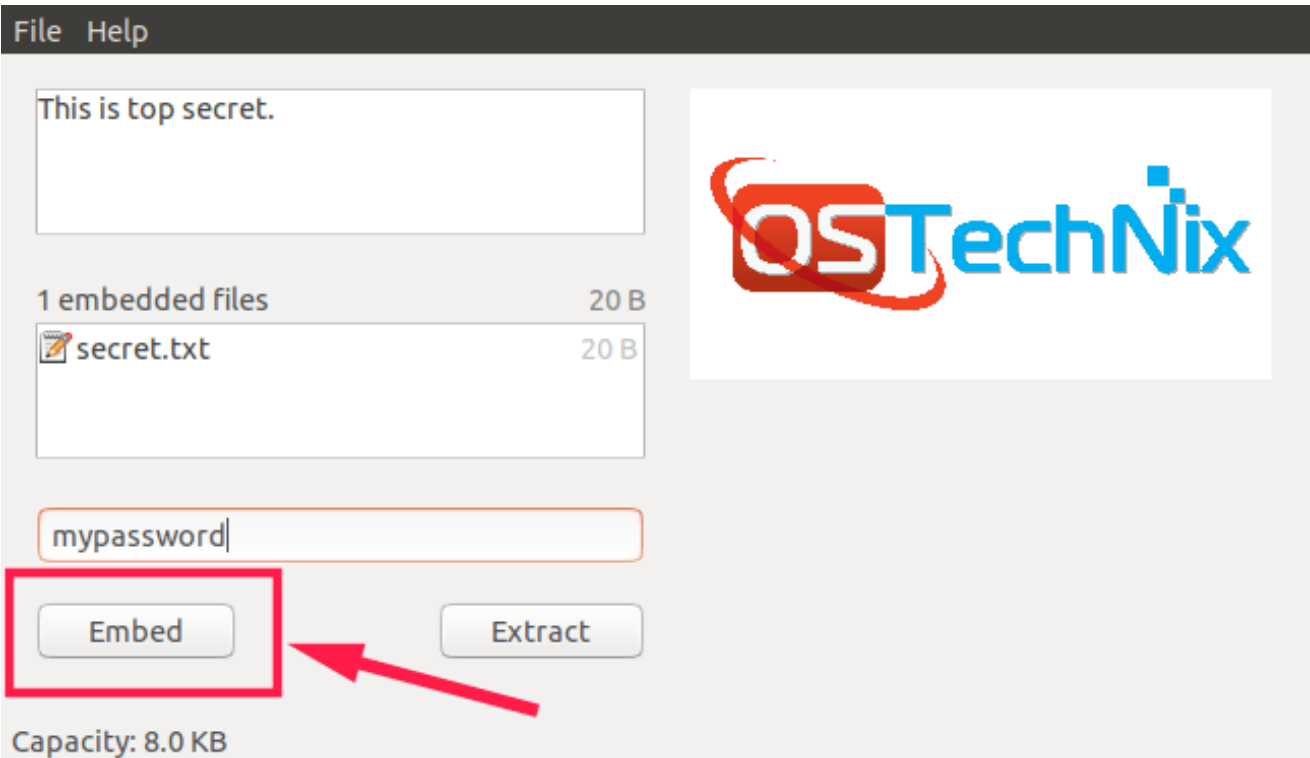
Stegosuite interface

Click **File -> Open** from the menu bar and choose an image that you want to use to hide the files. Then enter the message in the first column. Right click on the second column and choose "Add file..". Finally enter the passphrase on the third column.



Add files in stegosuite

Finally, click **Embed** button at the bottom of Stegosuite interface.



Embed files using stegosuite



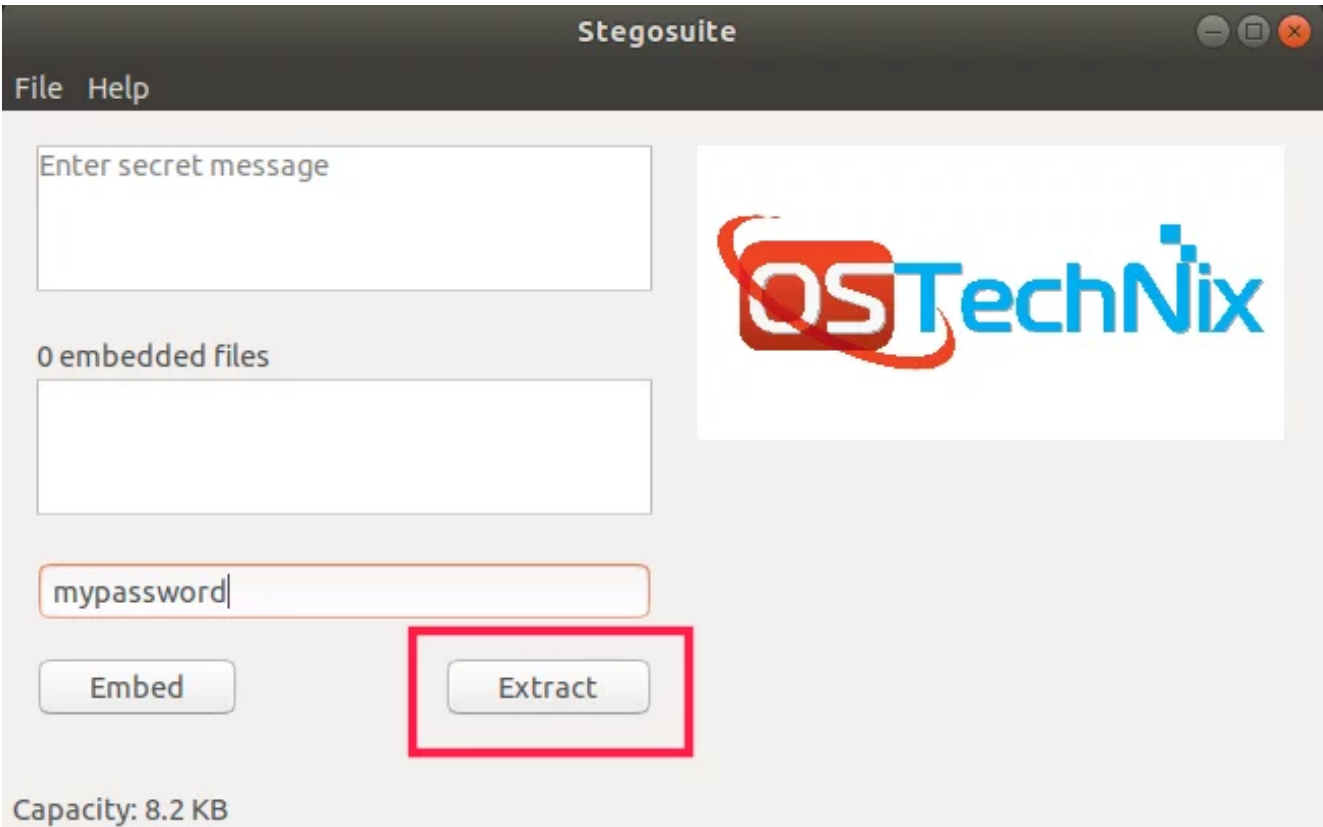
This website uses cookies to improve your experience. By using this site, we will assume that you're OK with it.

Accept

Read More

chosen the image from. For instance, if you have selected the image called "image.jpg" from Documents folder, a new image file will be created with name "image\_embed.jpg" in the Documents folder itself.

To extract the secret files from the image, just open it again in Stegosuite interface, enter the passphrase and click **Extract** button.



Extract files using stegosuite

All files will be extracted in the same folder itself.

For more details, refer [Stegosuite website](#).

## Method 5 - using Steg

**Steg** is a simple, cross platform, and graphical steganographic tool, written using **C++** programming language. It is a portable software, so just download it, carry it anywhere and start using it in no-time, regardless of any operating system you use.

Steg supports JPEG, JPG, TIFF, PNG, and BMP image formats. It uses Steganography and Cryptography techniques to hide data inside compressed or uncompressed images.

## Hide Files Inside Images With Steg

Click on [this link](#) to download the Steg application. It is available for both 32 and 64 bit architectures.

Or, Just use the following command to download it depending upon the architecture you use.

**For 64 bit:**

```
$ wget https://googledrive.com/host/0B-_yxJMDtRxyUExLZzZ3S2VDbjQ/steg-v1.0.0.2-linux64.tar.gz
```

**For 32 bit:**

```
$ wget https://googledrive.com/host/0B-_yxJMDtRxyRDNGNk1YcXR0UTg/steg-v1.0.0.2-linux32.tar.gz
```

After downloading, extract it using command:

```
$ tar -xvzf steg-v1.0.0.2-linux64.tar.gz
```

Go to Steg directory:



This website uses cookies to improve your experience. By using this site, we will assume that you're OK with it.

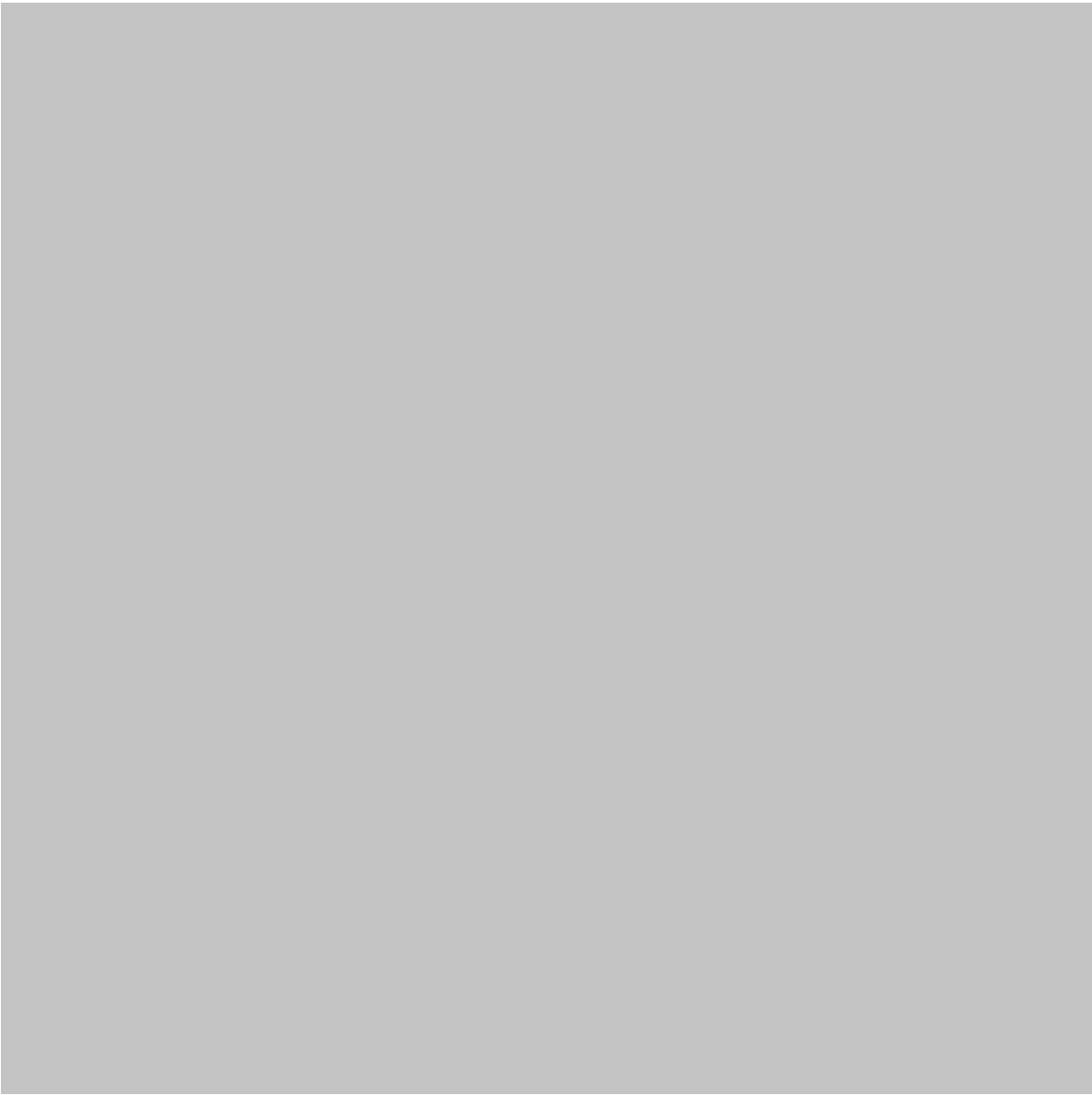
Accept

Read More

And, type the following command to run it.

```
$ ./steg.sh
```

Click **Yes** to accept the license agreement.



*Accept Eula*

Click Ok to continue.



*Steg Notification*

This is how Steg application default interface looks like.





*Steg Interface*

Now, let us hide some data inside an image.

To do that, go to **File -> Open generic image** or **Open JPEG image**. Make sure you have chosen **a big size image** to store more data inside of it. The bigger image you choose, the more you can save inside the image.

After you open the image, the original image and modified image (output image) will be shown in left and right panels respectively. Also, It displays the available size to store data inside the image in the bottom right corner.





Hide Files Inside Images In Linux Using Steg

Now, go to **Hide -> Hide Data** from the top menu bar. Select the file you want to hide. Make sure the file you selected is smaller than the available space in the modified image. After adding the data, you will see a confirmation dialog box that says: **Data successfully hidden.**



Data Is Hidden

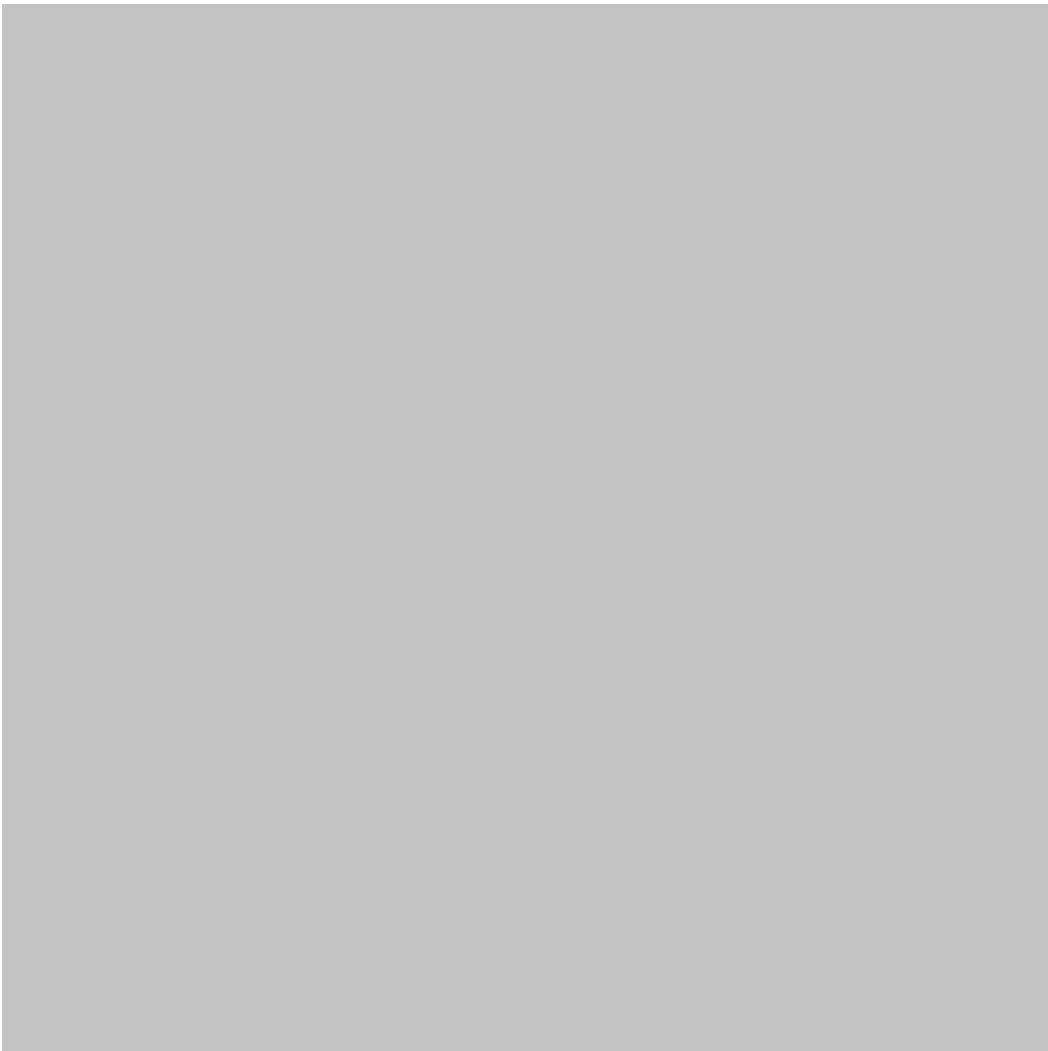
Here comes the important part. You can choose different encryption modes.

- **Auto:** Data will be encrypted, but you there is no PassPhrase or keys will be required to extract data.
- **Symmetric:** You have to give a PassPhrase to encrypt the data, and the recipient will need the same PassPhrase to extract it.
- **Asymmetric unsigned:** when you want to hide data (you are the sender) only the receiver's public key is required. When you want to extract data (you are the receiver) only your private key is required.
- **Asymmetric signed:** when you want to hide data (you are the sender) the receiver's public key and your private key are required. When you want to extract data (you are the receiver) only your private key is required but the sender's public key is requested. If you don't provide the sender's public key, at the end of the extraction process, you will be warned that the sender identity is not verified. If you provide the sender's public key you will be informed if sign verification is succeeded.



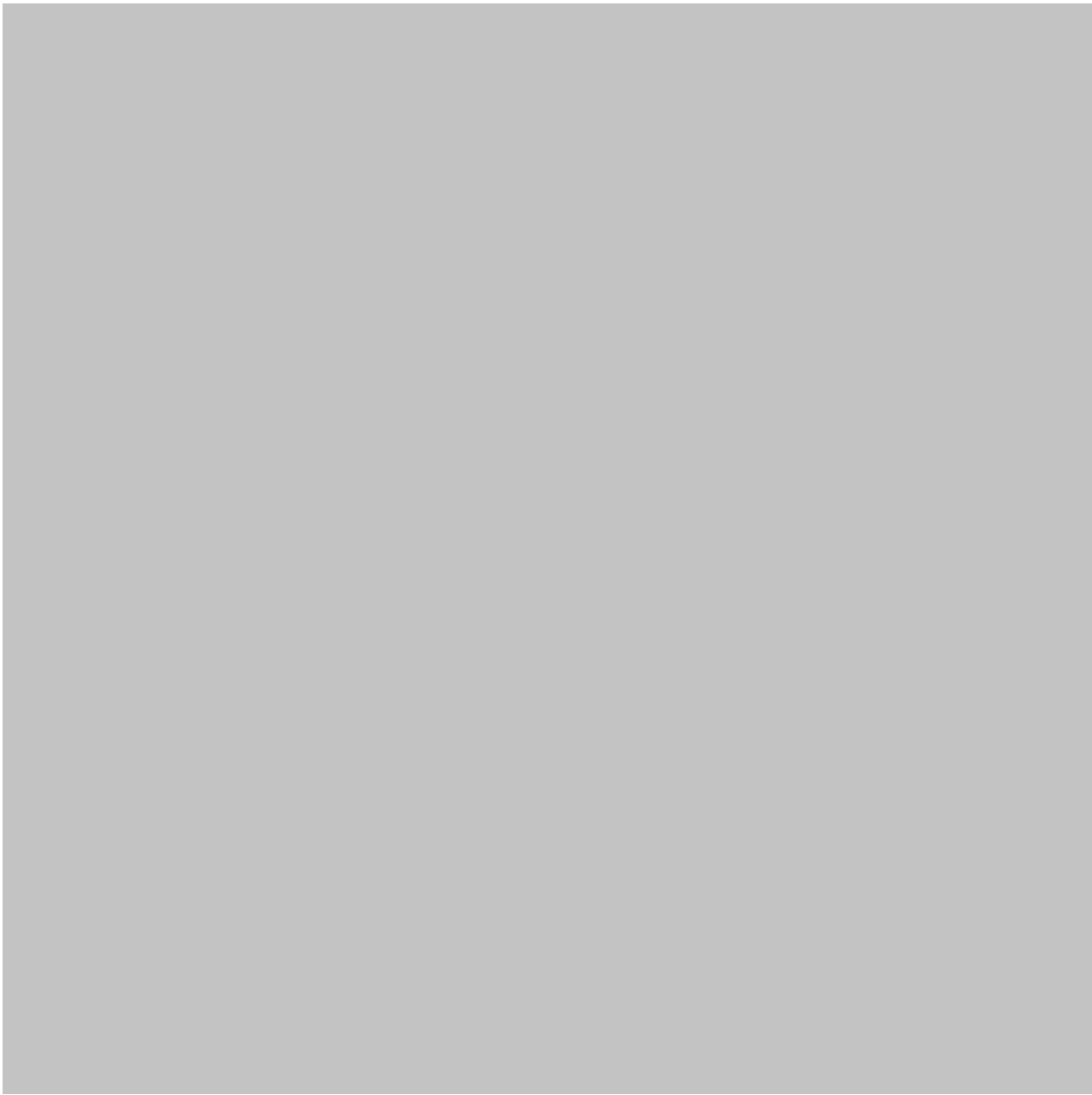
This website uses cookies to improve your experience. By using this site, we will assume that you're OK with it. [Accept](#) [Read More](#)

To choose a specific cryptography method, go to **Edit -> Configuration** from the menu bar. The default cryptography method is **auto**. Also, you can embed some messages on the file if you want to.



Steg Options

Once everything is ok, click the **Save** button on the tool bar, and save it in any location you prefer.



Save Media

Done! The image data is encrypted inside the image. This image will look like a regular image. You can view it using any image viewer application.

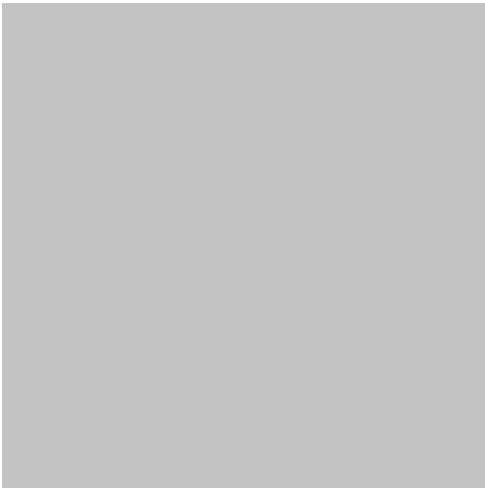
To extract the hidden data, just open the encrypted image in Steg application. To do so, go to **Extract -> Extract data** from the menu bar.



This website uses cookies to improve your experience. By using this site, we will assume that you're OK with it.

Accept

Read More




Data Extraction

That's it. You will now be able to view the data.


As you can see, this is extremely easy to follow and doesn't require any special skills. Just open an image, hide some confidential data, and pass it along.


For more details about Steg, check the [official website](#).


- HIDE FILES INSIDE IMAGES
- LINUX
- OUTGUESS
- SECURITY
- STEG
- STEGANOGRAPHY
- STEGHIDE
- STEGOSUITE


 3 comments


3










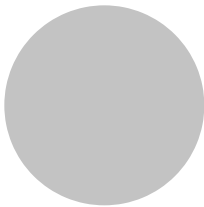
















SK


Senthilkumar Palani (aka SK) is the Founder and Editor in chief of OSTechNix. He is a Linux/Unix enthusiast and FOSS supporter. He lives in Tamilnadu, India.











Previous post

Next post

SSLH – Share A Same Port For HTTPS And SSH

How To Find Top Most Used Commands On Linux

YOU MAY ALSO LIKE


- KaliBrowser – Run Kali Linux Directly In Your...

June 28, 2016


Install Kali Linux Tools Using Katoolin In Ubuntu...

October 18, 2018

How To Use Ansible Vault To Protect Sensitive...

October 1, 2022
- 
- 3 COMMENTS
- 

TUTU VIP

 March 30, 2018 - 8:01 am

Thanks for the tip it worked.

REPLY
- EISSA

REPLY
- https://ostechnix.com/hide-files-inside-images-linux/

13/21

This website uses cookies to improve your experience. By using this site, we will assume that you're OK with it.

Accept

Read More

after using cat .. when i open the image the exe file wasn't excuted

EKA

🕒 August 16, 2019 - 11:07 am

Wow it worked using steghide

REPLY

LEAVE A COMMENT

Your Comment

Name\*

Email\*

☐ Save my name, email, and website in this browser for the next time I comment.

\* By using this form you agree with the storage and handling of your data by this website.

SUBMIT

This site uses Akismet to reduce spam. [Learn how your comment data is processed.](#)

Home > Secure Shell (SSH) > SSLH – Share A Same Port For HTTPS And SSH

SECURE SHELL (SSH) < COMMAND LINE UTILITIES < FAQ < LINUX < LINUX ADMINISTRATION < LINUX BASICS < LINUX COMMANDS < UBUNTU < UNIX/LINUX BEGINNERS < UTILITIES

SSLH – Share A Same Port For HTTPS And SSH

Written by Sk | **Published:** August 14, 2019 | **Last Updated on** August 28, 2022 | 13.1k views

💬 3 comments

1 ❤️

In this brief tutorial, we will see **what is SSLH**, how to install SSLH and how to configure SSLH to **share a same port for https and ssh** in Linux and Unix-like operating systems.

What is SSLH?

Some Internet service providers and corporate companies might have blocked most of the ports, and allowed only a few specific ports such as port 80 and 443 to tighten their security.

In such cases, we have no choice, but use a same port for multiple programs, say the HTTPS Port **443**, which is rarely blocked. Here is where **SSLH**, a SSL/SSH multiplexer, comes in help.

SSLH will listen for incoming connections on a port 443. To put this more simply, SSLH allows us to run several programs or services on port 443 on a Linux system. So, you can use both SSL and SSH using a same port at the same time.

If you ever been in a situation where most ports are blocked by the firewalls, you can use SSLH to access your remote server.



SSLH is packaged for most Linux distributions, so you can install it using the default package managers.

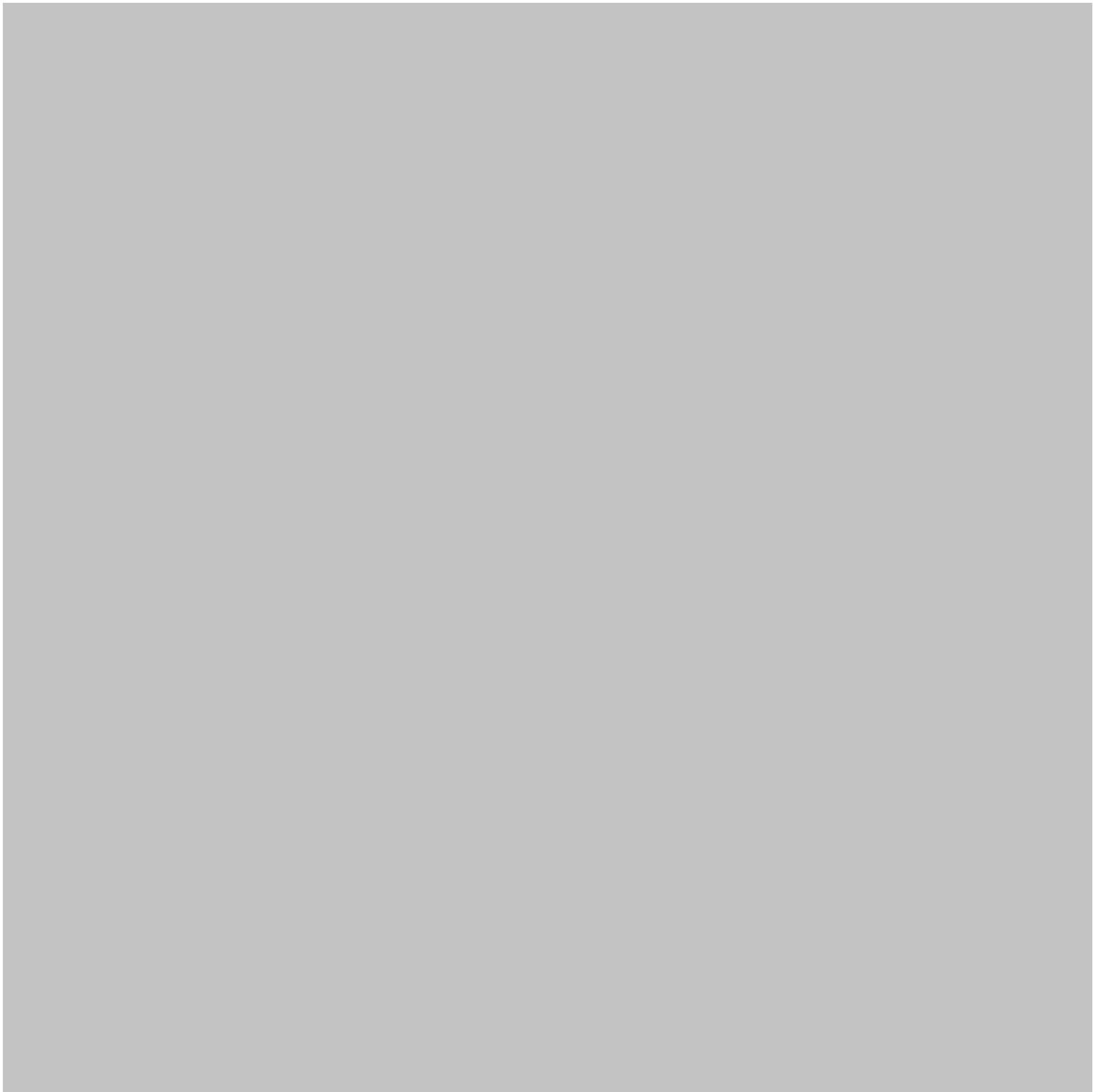
On **Debian**, **Ubuntu**, **Linux Mint** and **Pop OS**, run:

```
$ sudo apt install sslh
```

While installing SSLH, you will prompted whether you want to run sslh as a service from inetd, or as a standalone server.

Each choice has its own benefits. With only a few connection per day, it is probably better to run sslh from inetd in order to save resources.

On the other hand, with many connections, sslh should run as a standalone server to avoid spawning a new process for each incoming connection.



Install sslh

On **Arch Linux** and derivatives like Antergos, Manjaro Linux, install it using Pacman as shown below.

```
$ sudo pacman -S sslh
```

On **RHEL**, **CentOS**, **AlmaLinux** and **Rocky Linux**, you need to add **EPEL** repository and then install SSLH as shown below.

```
$ sudo dnf install epel-release
```



```
$ sudo dnf install sslh
```

On **Fedora**:

```
$ sudo dnf install sslh
```

If it is not available on default repositories, you can manually compile and install SSLH as described [here](#).

## Configure Apache or Nginx webserver

As you already know, Apache and Nginx webserver will listen on all network interfaces (i.e `0.0.0.0:443`) by default. We need to change this setting to tell the webserver to listen on the localhost interface only (i.e. `127.0.0.1:443` or `localhost:443`).

To do so, edit the webserver (nginx or apache) configuration file and find the following line:

```
listen 443 ssl;
```

And, change it to:

```
listen 127.0.0.1:443 ssl;
```

If you’re using Virtualhosts in Apache, make sure you have changed that it too.

```
VirtualHost 127.0.0.1:443
```

Save and close the config files. Do not restart the services. We haven't finished yet.

## Configure SSLH

Once you have made the webserver to listen on local interface only, edit SSLH config file:

```
$ sudo vi /etc/default/sslh
```

Find the following line:

```
Run=no
```

And, change it to:

```
Run=yes
```

Then, scroll a little bit down and modify the following line to allow SSLH to listen on port 443 on all available interfaces (Eg. `0.0.0.0:443`).

```
DAEMON_OPTS="--user sslh --listen 0.0.0.0:443 --ssh 127.0.0.1:22 --ssl 127.0.0.
```

This website uses cookies to improve your experience. By using this site, we will assume that you're OK with it. [Accept](#) [Read More](#)

Where,

- `--user sslh` : Requires to run under this specified username.
- `--listen 0.0.0.0:443` : SSLH is listening on port `443` on all available interfaces.
- `--sshs 127.0.0.1:22` : Route SSH traffic to port `22` on the localhost.
- `--ssl 127.0.0.1:443` : Route HTTPS/SSL traffic to port `443` on the localhost.

Save and close the file.

Finally, enable and start `sslh` service to update the changes.

```
$ sudo systemctl enable sslh
```

```
$ sudo systemctl start sslh
```

## Testing

Check if the SSLH daemon is listening to `443`.

```
$ ps -ef | grep sslh
sslh 2746 1 0 15:51 ? 00:00:00 /usr/sbin/sslh --foreground --user sslh --listen
sslh 2747 2746 0 15:51 ? 00:00:00 /usr/sbin/sslh --foreground --user sslh --lis
sk 2754 1432 0 15:51 pts/0 00:00:00 grep --color=auto sslh
```

Now, you can access your remote server via SSH using port `443`:

```
$ ssh -p 443 sk@192.168.225.50
```

### Sample output:

```
sk@192.168.225.50's password:
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-55-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Wed Aug 14 13:11:04 IST 2019

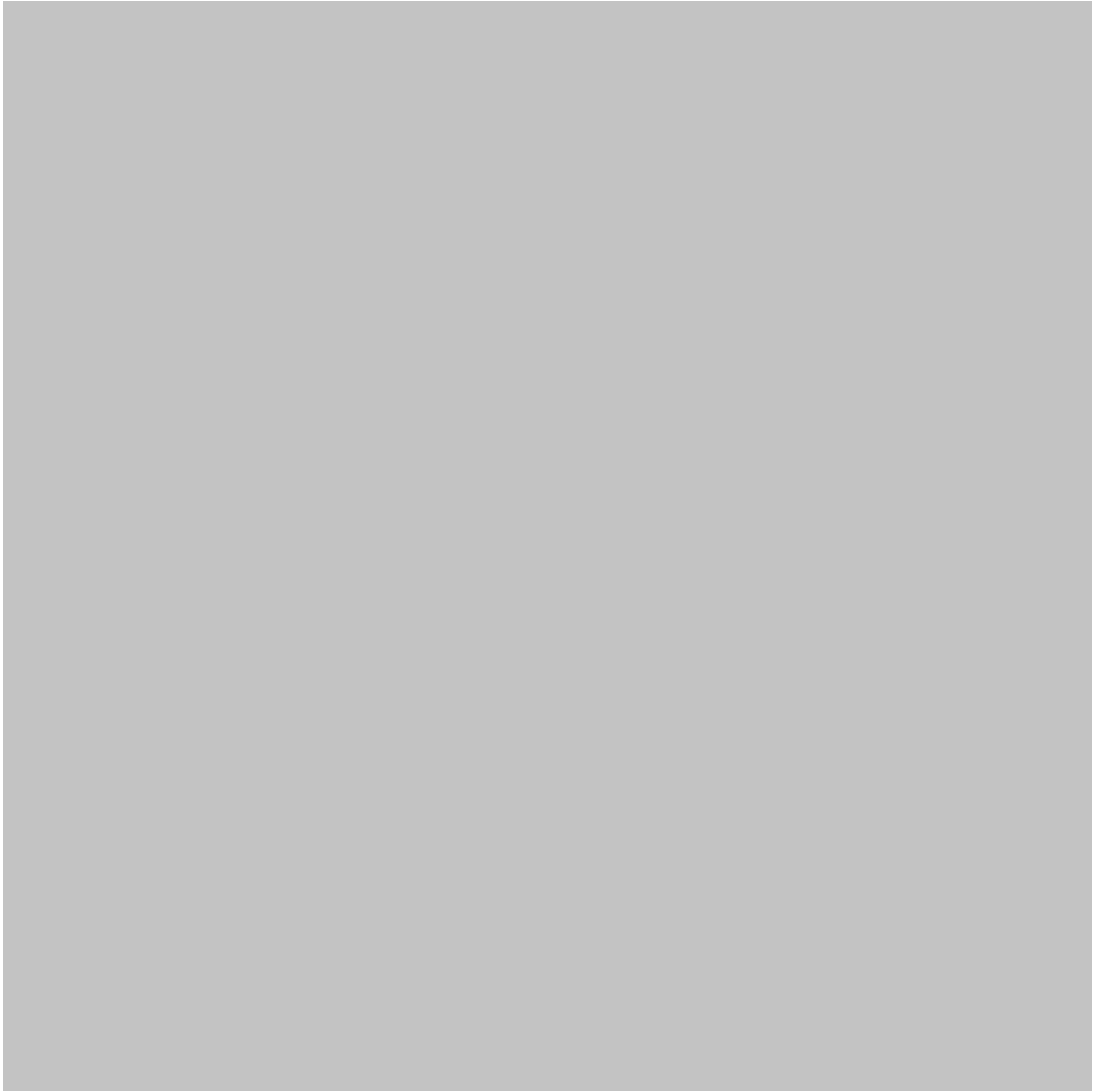
System load: 0.23 Processes: 101
Usage of /: 53.5% of 19.56GB Users logged in: 0
Memory usage: 9% IP address for enp0s3: 192.168.225.50
Swap usage: 0% IP address for enp0s8: 192.168.225.51

 * Keen to learn Istio? It's included in the single-package MicroK8s.

https://snapcraft.io/microk8s

61 packages can be updated.
22 updates are security updates.
```

Last login: Wed Aug 14 13:10:33 2019 from 127.0.0.1



Access remote systems via SSH using port 443

See? I can now be able to access the remote server via SSH even if the default SSH port 22 is blocked. As you see in the above example, I have used the https port 443 for SSH connection. Also, we can use the same port 443 for openVPN connections too.

## Conclusion

I tested SSLH on my Ubuntu 18.04 LTS server and it worked just fine as described above. I tested SSLH in a protected local area network, so I am not yet aware of the security issues. If you're using it in production, let us know the advantages and disadvantages of using SSLH in the comment section below.

For more details, check the official GitHub page given below.

### Resource:

- [SSLH GitHub Repository](#)

### Suggested read:

- [How To SSH Into A Particular Directory On Linux](#)
- [How To Create SSH Alias In Linux](#)
- [How To Configure SSH Key-based Authentication In Linux](#)
- [How To Stop SSH Session From Disconnecting In Linux](#)
- [Allow Or Deny SSH Access To A Particular User Or Group In Linux](#)



This website uses cookies to improve your experience. By using this site, we will assume that you're OK with it.

Accept

Read More

▪ ScanSSH – Fast SSH Server And Open Proxy Scanner

HTTPS

LINUX

SECURE CONNECTION

SHARE A SAME PORT FOR HTTPS AND SSH

SSH

SSL

SSLH

3 comments

1

♥

f

🐦

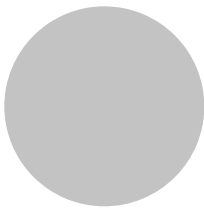
in

🍷

📞

📍

✉



SK

Senthilkumar Palani (aka SK) is the Founder and Editor in chief of OSTechNix. He is a Linux/Unix enthusiast and FOSS supporter. He lives in Tamilnadu, India.

🌐

f

🐦

in

📺

Previous post

Next post

Fix ‘E: The package cache file is corrupted, it has the wrong hash’ Error In Ubuntu

Steganography – Hide Files Inside Images In Linux

YOU MAY ALSO LIKE

How To Use sshpass For Non-interactive SSH login...

October 11, 2022


How To Setup Chrooted SFTP In Linux

September 9, 2021

How To Auto Logout Inactive Users After A...

September 18, 2021

3 COMMENTS




DANIEL ALEKSANDERSEN

REPLY

🕒 August 15, 2017 - 3:03 am

Okay, this is admittedly a clever work around if port availability is limited.. However, this sounds risky in terms of security. What new and interesting security issues may arise from a protocol multiplexer? Who knows? —and that right limits the usefulness of this until someone takes the time to audit the code.




ARUN KHAN

REPLY

🕒 August 17, 2017 - 11:55 pm

I connect to an openVPN server configured to run on 443 (login in with SSL cert + user auth). From there on I can connect to any of my servers over SSH. I have found this solution to work in many public WiFi hotspots. No experience with Enterprise networks, they may blacklist the IP numbers of public VPN providers.



TOMER GLICK

REPLY

🕒 January 31, 2018 - 12:29 am

That did not work for me. The reason is that by telling sslh to listen on 0.0.0.0:443, it listening on ALL interfaces including 127.0.0.1:443 which ssl already uses.

The work around I did is to use the actual IP of my server instead of 0.0.0.0. That option might not be

https://ostechnix.com/hide-files-inside-images-linux/

19/21

This website uses cookies to improve your experience. By using this site, we will assume that you're OK with it.

Accept

Read More

Something else than that.

LEAVE A COMMENT

Your Comment

Name\*

Email\*

☐ Save my name, email, and website in this browser for the next time I comment.

\* By using this form you agree with the storage and handling of your data by this website.

SUBMIT

This site uses Akismet to reduce spam. [Learn how your comment data is processed.](#)

ABOUT OSTECHNIX



OSTechNix (Open Source, Technology, Nix\*) regularly publishes the latest news, how-to articles, tutorials and tips & tricks about free and opensource software and technology.



Archives

Select Month

POPULAR POSTS

- 1

How From And ! Nover
- 2

How Error August
- 3

How Forw June 1



This website uses cookies to improve your experience. By using this site, we will assume that you're OK with it. [Accept](#) [Read More](#)