PGP SKS-Keyserver

< [PGP](https://wiki.mattrude.com/PGP)‎ | [Keyserver](https://wiki.mattrude.com/PGP/Keyserver)

[Jump to navigation](https://wiki.mattrude.com/PGP/Keyserver/SKS#mw-head)[Jump to search](https://wiki.mattrude.com/PGP/Keyserver/SKS#p-search)

noteThis Page was written with [**Ubuntu**](https://wiki.mattrude.com/Ubuntu)**14.04 LTS** in mind, and may not work correctly with other versions or distributions.

A Key Server is used to distribute [PGP](https://wiki.mattrude.com/PGP) keys between different users. One of the most popular key servers for use with pgp/gpg is the [sks key-server](https://sks-keyservers.net/). This document will walk you through downloading, installing, and setting up a sks key-server on [Ubuntu](https://wiki.mattrude.com/Ubuntu) 14.04 LTS.



**Contents**

* [1Building the SKS Daemon](https://wiki.mattrude.com/PGP/Keyserver/SKS#Building_the_SKS_Daemon)
* [2Configure the sks-server](https://wiki.mattrude.com/PGP/Keyserver/SKS#Configure_the_sks-server)
* [3Download the needed database files](https://wiki.mattrude.com/PGP/Keyserver/SKS#Download_the_needed_database_files)
  + [3.1Download Keydump](https://wiki.mattrude.com/PGP/Keyserver/SKS#Download_Keydump)
* [4Import the downloaded databases files](https://wiki.mattrude.com/PGP/Keyserver/SKS#Import_the_downloaded_databases_files)
* [5Configure your web-server](https://wiki.mattrude.com/PGP/Keyserver/SKS#Configure_your_web-server)
* [6Start the SKS Daemon](https://wiki.mattrude.com/PGP/Keyserver/SKS#Start_the_SKS_Daemon)
* [7Patches for the sks-keyserver software](https://wiki.mattrude.com/PGP/Keyserver/SKS#Patches_for_the_sks-keyserver_software)
  + [7.1SKS-Keyserver v1.1.5](https://wiki.mattrude.com/PGP/Keyserver/SKS#SKS-Keyserver_v1.1.5)

Building the SKS Daemon

The following is for [Ubuntu](https://wiki.mattrude.com/Ubuntu) 14.04 LTS

apt-get -y install gcc ocaml libdb6.0-dev gnupg wget

After installing the required software, you need to download SKS

gpg --keyserver hkp://pool.sks-keyservers.net --trust-model always --recv-key 0x0B7F8B60E3EDFAE3

wget https://bitbucket.org/skskeyserver/sks-keyserver/downloads/sks-1.1.5.tgz

wget https://bitbucket.org/skskeyserver/sks-keyserver/downloads/sks-1.1.5.tgz.asc

gpg --keyid-format long --verify sks-1.1.5.tgz.asc

The output of the last command should be

gpg: Signature made Mon 05 May 2014 02:06:51 PM CDT

gpg: using RSA key 41259773973A612A

gpg: Good signature from "SKS Keyserver Signing Key"

Now, untar the software

tar -xzf sks-1.1.5.tgz

cd sks-1.1.5

Next copy the **Makefile.local.unused** to **Makefile.local** and change ldb-4.6 to ldb-6.0 for Ubuntu.

cp Makefile.local.unused Makefile.local

sed -i 's/ldb\-4.6/ldb\-6.0/' Makefile.local

Last, build the software

make dep

make all

make install

Configure the sks-server

**/var/lib/sks/sksconf**

# /var/lib/sks/sksconf

debuglevel: 3

# Set the hostname of your server

hostname: --keyserver-hostname--

hkp\_address: 127.0.0.1

hkp\_port: 11371

recon\_port: 11370

# Set the PGP ID for the Server Contact

server\_contact: --contact-pgp-id--

initial\_stat:

disable\_mailsync:

membership\_reload\_interval: 1

stat\_hour: 12

max\_matches: 500

Download the needed database files

Rather than starting with an empty database and attempting to populate it by syncing with other keyservers (a bad idea because it loads up your peers with lots of traffic and will probably fail anyway with deadlocks in the conflict resolution system) we'll grab a static dump from an existing SKS server. Currently the only known source is:

* <http://keyserver.mattrude.com/dump/> - Generated every DAY
* <ftp://ftp.prato.linux.it/pub/keyring/> - Generated every Wednesday
* <http://keyserver.borgnet.us/dump> - Generated every Sunday

**Download Keydump**

The keydump is about 6.3GB as of Oct 2014, so fetching it will take a long time. It's divided into a bunch of individual numbered files so you'll need to fetch all of them. Because I'm too lazy to spend 8 hours sitting there doing it manually I did it like this:

mkdir /var/lib/sks/dump

cd /var/lib/sks/dump

wget -c -r -p -e robots=off --timestamping --level=1 --cut-dirs=3 \

--no-host-directories http://keyserver.mattrude.com/dump/current/

Many hours later, check that all the pieces downloaded correctly by comparing their checksums against the list published by the dump provider:

md5sum -c metadata-sks-dump.txt

Import the downloaded databases files

There are two ways to do this: either a full build (which reads in the dump you just downloaded and leaves you with a complete, self-contained database) or a fastbuild (which just references the dump and requires it to be left in place after the fastbuild is complete). I started doing a full build, it looked like it was going to take forever so I aborted it and switched to a quickbuild. On the 4-processor machine I was using it still took in the order of 40 minutes to run so this might take a while.

You need to be in the basedir when running this and the dumps have to be in a sub-directory called dump (which they should be if you followed the steps above), so:

cd /var/lib/sks

/usr/local/bin/sks\_build.sh

On the next screen, choose **2**.

Please select the mode in which you want to import the keydump:

1 - fastbuild

only an index of the keydump is created and the keydump cannot be

removed.

2 - normalbuild

all the keydump will be imported in a new database. It takes longer

time and more disk space, but the server will run faster (depending

from the source/age of the keydump).

The keydump can be removed after the import.

Enter enter the mode (1/2): 2

If you edit the sks\_build.sh script you’ll discover it’s just a shell script which calls SKS itself to do the heavy lifting. If you have trouble with lack of memory you may need to tweak the script a bit: in particular the -n 10 flag used in the fastbuild call is a multiple of 15,000 keys to load at a time. The default setting therefore loads 150,000 keys at a time which could cause your machine to go into swap, and changing to something like -n 2 will cause it to load only 30,000 at a time instead and possibly complete the job faster. The trick is to load as many as possible in each pass without hitting swap - if that happens, performance falls through the floor and you may as well abort it and start again (after deleting the KDB and PTree directories created by the aborted import).

If all goes smoothly you'll end up with KDB and PTree directories in /var/lib/sks.

Configure your web-server

**/etc/nginx/nginx.conf**

#/etc/nginx/nginx.conf

user www-data;

worker\_processes 4;

pid /run/nginx.pid;

events {

worker\_connections 768;

}

http {

sendfile on;

tcp\_nopush on;

tcp\_nodelay on;

client\_max\_body\_size 8m;

access\_log /var/log/nginx/access.log;

error\_log /var/log/nginx/error.log;

rewrite\_log on;

include /etc/nginx/mime.types;

#----------------------------------------------------------------------

# OpenPGP Public SKS Key Server

#----------------------------------------------------------------------

server {

listen 80 default\_server;

listen [::]:80 default\_server;

listen --IPv4-Address-- default\_server;

listen [--IPv6-Address--]:11371 default\_server;

server\_name \*.sks-keyservers.net;

server\_name \*.pool.sks-keyservers.net;

server\_name pgp.mit.edu;

server\_name keys.gnupg.net;

root /var/www/html;

rewrite ^/stats /pks/lookup?op=stats;

rewrite ^/s/(.\*) /pks/lookup?search=$1;

rewrite ^/search/(.\*) /pks/lookup?search=$1;

rewrite ^/g/(.\*) /pks/lookup?op=get&search=$1;

rewrite ^/get/(.\*) /pks/lookup?op=get&search=$1;

rewrite ^/d/(.\*) /pks/lookup?op=get&options=mr&search=$1;

rewrite ^/download/(.\*) /pks/lookup?op=get&options=mr&search=$1;

location /pks {

proxy\_pass http://127.0.0.1:11371;

proxy\_pass\_header Server;

add\_header Via "1.1 --keyserver-hostname--:11371 (nginx)";

proxy\_ignore\_client\_abort on;

client\_max\_body\_size 8m;

}

}

}

Start the SKS Daemon

**/etc/init.d/sks**

#! /bin/sh DAEMON=/usr/local/bin/sks

DIR=/var/lib/sks

test -e $DAEMON || exit 0

test -d $DIR || exit 0

case "$1" in

start) cd $DIR echo -n "Starting SKS:" echo -n \ sks\_db

$DAEMON db &

echo -n \ sks\_recon

$DAEMON recon &

echo "." ;;

stop) echo -n "Stopping SKS:"

killall sks

while [ "`pidof sks`" ]; do sleep 1; done # wait until SKS processes have exited echo "." ;;

restart) $0 stop

sleep 1

$0 start

;; \*) echo "Usage: $0 {start|stop|restart}" exit 1

;; esac exit 0

Patches for the sks-keyserver software

**SKS-Keyserver v1.1.5**

* [sks-1.1.5-download-txt.patch](https://gist.github.com/mattrude/709b2726ceb9d2d3386b) - Patch to change the downloadable “pgpkey.asc” file to a txt file, viewable in a web browser. ([example](http://keyserver.mattrude.com/pks/lookup?search=0x27143affdd23bf73&options=mr&op=get)) - [download](https://gist.githubusercontent.com/mattrude/709b2726ceb9d2d3386b/raw/sks-1.1.5-download-txt.patch)

[Categories](https://wiki.mattrude.com/Special:Categories):

* [Ubuntu](https://wiki.mattrude.com/Category:Ubuntu)
* [Ubuntu v14.04 LTS](https://wiki.mattrude.com/Category:Ubuntu_v14.04_LTS)
* [Email](https://wiki.mattrude.com/Category:Email)