# KeysAtApache

**Contents**

## Overview

When releasing software in an Apache project, the Release Manager must sign the jars and other artifacts with a private pgp key. Using the signer's public key, users can verify that signed artifacts were built by the signer.

Software is available that creates pgp public/private key pairs, signs keys and other files, encrypts data, and maintains key rings (files of private and public keys). The instructions below describe the use of GnuPg. See More information for links to other key software.

See the GnuPg FAQ How does this whole thing work? for a brief overview of pgp keys.

## Creating a key and having it signed

Follow these steps to obtain a key and have it signed. Detailed instructions follow.

1. Download GnuPg for creating and signing keys.
2. Create your key. Note your key fingerprint and ID (this is the last 8 digits of the fingerprint). You will need them later.
3. Generate a revocation certificate.
4. Upload your key to a public key server.
5. Publish your key at Apache.
6. If you are a committer, put your public key in $HOME/.pgpkey.
7. Contact people to sign your key.
8. Sign other people's keys.

### Using gpg to create a key

Always use a private, secure machine to create your key, because that is where your private key is. Never use a machine in the Apache infrastructure and never store your private key on such a machine.

```
gpg --gen-key
```
In response to the prompts,

- select DSA and ElGamal
- select key size, typically 1024 for DSA and 2048 or 4096 for ElGamal (the more bits, the more secure)
- select expiration date, typically 0 = key does not expire
- provide a user ID (your name, a comment, and your email address)
- enter a passphrase (should be long and it is important not to forget it)

Gpg will return your key id on a line like this:

pub 2048D/29D21F35 2013-06-27
Key fingerprint = 7FA7 3D8E B51C 909E C0C1 802F 43D1 E7DE 29D2 1F35

The eight digit hex number after "2048D/" is the key id.

For more information, see Getting Started in the GNU Privacy Manual.

### Generating a revocation certificate

```
gpg --output revoke.asc --gen-revoke key_id
```

The certificate in revoke.asc may be printed out and kept in a very safe place. You can use it to revoke a key even if you have lost your key or pass phrase.

### Uploading your public key

The MIT public key server is commonly used and provides a web form for uploading your key. You may also use the following command to upload keys:
`gpg --send-keys --keyserver pgp.mit.edu <key_id>`

### Publishing your key at Apache

Publish the public half of your key for verifying releases you sign by adding your key's fingerprint to your Apache LDAP profile (https://id.apache.org/).

### Finding people to sign your key

A key without signatures has no value; it could belong to anyone. One signature is better than none, but more is better and it is best to have some signatures within the Apache web of trust. You should also sign other people's keys.

- Individually

Anyone who knows you personally and has a key can sign your key. You need to provide them with your key fingerprint and owner information, which you get by this command:

```
gpg --fingerprint KEY_ID
```

You can use Biglumber to find people to exchange signatures with.

- You can find lots of people to sign your key at an Apache key signing party

Some people may email you your signed key rather than uploading it to a public server. If so, just import it and upload it yourself.

### Signing a key

1. Import the person's public key from pgp.mit.edu:
   `gpg --keyserver pgp.mit.edu --recv-keys <key_id>`
   or, if you have received a key file, import the keys from the file:
   `gpg --import <key_file>`
2. Verify the fingerprint – does it exactly match the hardcopy from the ApacheCon key signing?
   `gpg --fingerprint <owner_email`
3. Sign the key:
   `gpg --sign-key <key_id>`
4. Upload the signed key:
   `gpg --send-keys --keyserver pgp.mit.edu <key_id>`
   or, export it and email it to the owner to upload:
   `gpg --armor --export <owner_email> > ownerkeyid_signed_by_keyid`
   or
   `gpg --armor --export <key_id> > ownerkeyid_signed_by_keyid`

## Signing a release with your key

- Publish your key to the KEYS file
- Sign the release to create a detached signature file
- Post the release and its signature to the distribution directory
- Add a checksum file to the dist directory (optional)
- Point to instructions on how to verify signatures

See http://www.apache.org/dev/release-signing.html for more information.

## Verifying a signed release

Unless you verify the integrity of downloaded files using the PGP signature and/or the MD5 checksum, you cannot be sure of their authenticity. The checksum is not as strong an indicator as the PGP signature is.

The PGP signatures can be verified using PGP or GPG. First download the KEYS as well as the .asc signature file for the particular distribution. Make sure you get the KEYS and signatures from the main distribution directory, rather than from a mirror. Then verify the signatures using

```
gpg --import KEYS

gpg --verify release_name.tar.gz.asc
```

You can also verify the checksums on the files. Unix programs called md5/sha1 or md5sum/sha1sum are included in many unix distributions. *sum is also available as part of GNU Textutils. Windows users can get binary md5 programs from http://www.fourmilab.ch/md5 and hhttp://www.pc-tools.net/win32/freeware/console. Windows SlavaSoft fsum supports MD5 and SHA1. You can also use the web form at http://people.apache.org/~henkp/cgi-bin/md5.cgi to verify a checksum. It is best to verify the PGP signature, though. The signature verifies both the integrity of the file and the identity of the person who published the release.

## More information

- Documentation for GnuPG
  - The GNU Privacy Handbook
- Other pgp software
  - PGP 8.0 Freeware
  - PGP 8.0 (commercial)
- List and web map of Apache committers' keys
- The Apache release signing policy.
- A cool tool for viewing trust paths