

关键一代

生成GPG签名密钥对

本指南将介绍如何生成新的密钥对，如何创建签名子密钥以及如何剥离密钥主密钥，以避免您的主密钥身份泄漏，以防您的签名密钥（或整个 ~/.gnupg/ ）丢失。

1) 在安全机器上生成新的密钥对

```
$ mkdir / tmp / 签名
$ chmod 0700 / tmp / 签
$ gpg --homedir / tmp / 签名 --gen-key
gpg (GnuPG) 1.4.18; 版权所有©2014 Free Software Foundation, Inc.
这是免费软件：您可以随意更改并重新分发。
在法律允许的范围内，不作任何保证。
```

```
gpg: keyring'/tmp/signing/secring.gpg'创建
gpg: keyring'/tmp/signing/pubring.gpg'创建
```

小 费	选择4以生成仅限RSA的密钥，并选择4096位的密钥大小。对于这个我如何选择不设定任何到期。
--------	--

请选择你想要的那种密钥：

- (1) RSA和RSA（默认）
- (2) DSA和Elgamal
- (3) DSA（仅限符号）
- (4) RSA（仅限符号）

你的选择？ 4

RSA密钥可以在1024到4096位之间。

你想要什么键 （2048） 4096

请求的密钥大小为4096位

请指定密钥应该有效的时间。

0 =键不会过期

<n> =密钥在n天过期

<n> w =密钥在n周内到期

<n> m =密钥在n个月内到期

<n> y =密钥在n年内到期

钥匙是否适用？（0）

钥匙根本不到期

它是否正确？（y / N） y

小	GPG现在将询问您的用户身份，提供您打算用于项目沟通的真实姓名和邮箱地址。我也建议
---	---

费 提供一个有意义的评论，例如“LEDE签名”

您需要一个用户ID来标识您的密钥；该软件以此形式从实名，评论和电子邮件地址构建用户ID：
“Heinrich Heine (Der Dichter) ”heinrichh@duesseldorf.de“”

真名：乔·菲利普

电子邮件地址：<jo@mein.io>

评论：LEDE签名

您选择了此USER-ID:

“Jo-Philipp Wich (LEDE签字) ”jo@mein.io >>“

更改 (N) ame, (C) omment, (E) 邮件或 (O) kay / (Q) uit? Ø

你需要一个密码来保护你的密钥。

小 在这一点上，输入一个好的密码两次来保护你的秘密密钥，命令将需要一段时间来收集熵和
费 完整的密钥，直到最终打印出关键的摘要：

```
gpg: /tmp/signing/trustdb.gpg: 创建trustdb
gpg: 键612A0E98标记为最终被信任的公钥和密钥创建并签名。
```

```
gpg: 检查trustdb
```

```
gpg: 需要3个边缘，需要1个完整的PGP信任模型
```

```
gpg: depth: 0 valid: 1 signed: 0 trust: 0-, 0q, 0n, 0m, 0f, 1u
```

```
酒吧4096R / 612A0E98 2016-04-05
```

```
    钥匙指纹= 69B2 6A27 62D0 65E6 6F59 6755 C76F DE50 612A 0E98
```

```
ui Jo-Philipp Wich (LEDE签名) << jo@mein.io >>
```

请注意，此密钥不能用于加密。你可能想使用

命令“--edit-key”为此生成一个子项。

2) 生成子键

```
$ gpg --homedir / tmp /签名--edit-key <jo@mein.io>
```

```
gpg (GnuPG) 1.4.18; 版权所有©2014 Free Software Foundation, Inc.
```

这是免费软件：您可以随意更改并重新分发。

在法律允许的范围内，不作任何保证。

秘密密钥可用。

```
pub 4096R / 612A0E98创建时间: 2016-04-05 expires: never usage: SC
```

```
    信任: 终极有效性: 终极
```

```
[终极] (1)。Jo-Philipp Wich (LEDE签名) << jo@mein.io >>
```

小 在现在出现的交互式gpg提示符中输入“addkey”来创建一个新的签名子项。GnuPG将使用您
费 在上一步中给出的密码短语来解锁主密钥。

```
gpg> addkey
```

密钥被保护。

您需要密码才能解锁密钥

用户: “Jo-Philipp Wich (LEDE签名)”jo@mein.io >>“

4096位RSA密钥, ID 612A0E98, 创建于2014-04-05

请选择你想要的那种密钥:

- (3) DSA (仅限符号)
- (4) RSA (仅限符号)
- (5) Elgamal (仅限加密)
- (6) RSA (仅加密)

小费 我们将使用仅签名的4096位RSA密钥, 有效期为两年

你的选择? 4

RSA密钥可以在1024到4096位之间。

你想要什么键 (2048) 4096

请求的密钥大小是4096位请指定密钥应该有效多长时间。

0=键不会过期

<n> =密钥在n天过期

<n> w =密钥在n周内到期

<n> m =密钥在n个月内到期

<n> y =密钥在n年内到期

钥匙是否适用? (0) 730

钥匙到期4月5日18:19:42 2018 CEST

它是否正确? (y / N) y

真的创造? (y / N) y

小费 在这一点上, GnuPG将再次开始收集熵, 在后台运行“find /”是加快速度的好办法。完成后, 它将打印子键摘要并返回到提示。注意子项的ID“1584F206”, 我们需要在步骤4中。

```
pub 4096R / 612A0E98创建时间: 2016-04-05 expires: never usage: SC
```

信任: 终极有效性: 终极

```
sub 4096R / 1584F206创建时间: 2016-04-05过期日期: 2018-04-05用法: S
```

```
[终极] (1)。Jo-Philipp Wich (LEDE签名) << jo@mein.io >>
```

小费 输入“保存”提交新的密钥对其子密钥到磁盘, GnuPG将自动退出到外壳。

```
gpg>保存
```

3) 将钥匙放入保险库

在这一点上, 对目录进行可靠和安全的备份是一个好主意 /tmp/signing/, 我建议将其刻录到CDROM上或将其复制到拇指驱动器上, 您可以安全地将其锁定或隐藏在您的公寓中:)

4) 仅导出私有子键

我们现在将导出秘密子密钥，因为我们需要签署文件。使用步骤2中的子密钥ID，然后使用感叹号选择要导出的子密钥：

```
$ gpg --homedir / tmp / 签名 --export-secret-subkeys 1584F206! \  
  > /tmp/secret-signing-key.gpg  
$ file /tmp/secret-signing-key.gpg  
secret-signing-key.gpg: PGP \ 011Secret Key - 1024b创建于星期二4月5日16:08:15 2016  
- RSA (加密或签名)
```

5) 将秘密签名子密钥导入实际密钥存储区

您现在可以在将来用于签名文件的任何计算机上导入密码签名子密钥。要导入子密钥文件，请将其传递给 `gpg --import` 另一个 `homedir` 参数：

```
$ gpg --import /tmp/secret-signing-key.gpg  
gpg: 键612A0E98: 导入密码  
gpg: key 612A0E98: 公钥“Jo-Philipp Wich (LEDE签名密钥)”jo@mein.io >>“导入  
gpg: 处理的总数: 1  
gpg: imported: 1 (RSA: 1)  
gpg: 密钥读取: 1  
gpg: 进口密钥: 1
```

小 您现在可以发出一个“`gpg -K`”来列出密钥库中的所有密钥，您应该看到导入“密码”的密钥。这
费 里的哈希标记表示秘密主密钥丢失，这是我们想要的。

```
$ gpg -K  
/home/jow/.gnupg/secring.gpg  
-----  
[...]  
秒#4096R / 612A0E98 2016-04-05  
ui Jo-Philipp Wich (LEDE签名) << jo@mein.io >>  
ssb 4096R / 1584F206 2016-04-05
```

小 现在是将您的公共密钥部分上传到密钥服务器的时候了，以便其他人可以通过其指纹或您以后
费 选择的邮件地址轻松获取它。对于上传，使用上一个命令中“`sec#`”字之后打印的主键ID。

```
$ gpg --keyserver hkp: %% // %% pool.sks-keyservers.net --send-keys 612A0E98  
gpg: 将键612A0E98发送到hkp服务器pool.sks-keyservers.net
```

6) 删除原件

确保您的目录备份 `/tmp/signing` 完整可读，然后删除秘密子密钥文件和整个临时签名目录：

```
$ rm -r / tmp / signature /  
$ rm /tmp/secret-signing-key.gpg
```

小 您现在已经完成了一个合适的签名密钥对的设置。

7) 完成

要以ASCII (American Standard Code for Information Interchange)格式导出公钥，请使用以下命令，再次使用已用于上传pubkey的主ID。

确保提供有意义的评论，以便人们查看密钥文件知道其属于谁，而不必使用GPG实用程序检查它：

```
$ gpg --armor --export --no-version \  
    --comment =“Jo-Philipp Wich的公钥”612A0E98
```

要使用您的签名子密钥签名文件，请使用以下命令：

```
$ gpg --no-version -a -b -u 612A0E98 \  
    --comment =“我的签名的东西”-o output.sig input.file
```

小费 将公用签名密钥添加到存储库时，使用密钥ID作为文件名：

```
$ cd keyring / gpg /  
$ gpg --armor --export --no-version \  
    --comment =“我自己的公钥”612A0E98> 612A0E98.asc  
$ git add 612A0E98.asc  
$ git commit -sm“添加我的公钥”  
$ git push origin master
```

生成用户名密钥对

为了生成用于LEDE版本和软件包仓库的用户名密钥对，请按照以下步骤操作。

1) 取得用户权限

克隆usign存储库并进行编译。请注意，编译需要安装cmake才能成功。

```
$ git clone https://git.openwrt.org/project/usign.git  
$ cd usign /  
$ cmake  
$ make
```

小费 运行 ./usign 以检查二进制文件是否工作。

```
$ ./usign用法: ./usign <command> <options>
```

命令:

- V: **verify** (至少需要-m和-p | -P)
- S: 符号 (至少需要-m和-s)
- F: 公钥/秘密密钥或签名的打印密钥指纹
- G: 生成一个新的关键字

选项:

- c <comment>: 向键添加注释
- m <file>: 消息文件
- p <file>: 公钥文件 (仅验证/指纹)
- P <path>: 公钥目录 (仅验证)
- q: 安静 (不打印验证结果, 仅使用返回码)
- s <file>: 密钥文件 (仅限签名/指纹)
- x <file>: 签名文件 (默认为<message file> .sig)

2) 生成密钥对

指示 `usign` 可执行文件生成一个新的密钥对, 并提供适当的注释, 以便以后能够识别密钥文件。

```
./usign -G -c“LEDE用于Jo-Philipp的钥匙”  
-s secret.key -p public.key
```

小费 将 `secret.key` 文件存储在安全可靠的位置, 您将需要它来签署软件包存储库。


3) 将公钥添加到存储库

`usign -F` 使用命令 获取您的公钥的指纹, 并将其作为文件名用于将 `pubkey` 存储在存储 `keyring.git` 库中:

```
$ ./usign -F -p public.key  
72a57f2191b211e0
```

小费 将密钥添加到Git, 使用指纹作为文件名:

```
$ cd keyring / usign /  
$ cp /some/where/public.key 72a57f2191b211e0  
$ git add 72a57f2191b211e0  
$ git commit -sm“添加我的公共用户名”  
$ git push origin master
```

 最后修改: 2016/10/23 22:43 通过bobafetthotmail

除非另有说明, 本维基的内容将根据以下许可证获得许可: [CC Attribution-Share Alike 4.0 International](http://creativecommons.org/licenses/by-sa/4.0/)
(<http://creativecommons.org/licenses/by-sa/4.0/>)

