NAME

gpg-wks-server - Server providing the Web Key Service

SYNOPSIS

gpg-wks-server [*options*] **--receive**
gpg-wks-server [*options*] **--cron**
gpg-wks-server [*options*] **--list-domains**
gpg-wks-server [*options*] **--check-key** *user-id*
gpg-wks-server [*options*] **--install-key** *file user-id*
gpg-wks-server [*options*] **--remove-key** *user-id*
gpg-wks-server [*options*] **--revoke-key** *user-id*

DESCRIPTION

The **gpg-wks-server** is a server side implementation of the Web Key Service. It receives requests for publication, sends confirmation requests, receives confirmations, and published the key. It also has features to ease the setup and maintenance of a Web Key Directory.

When used with the command **--receive** a single Web Key Service mail is processed. Commonly this command is used with the option **--send** to directly send the created mails back. See below for an installation example.

The command **--cron** is used for regular cleanup tasks. For example non-confirmed requested should be removed after their expire time. It is best to run this command once a day from a cronjob.

The command **--list-domains** prints all configured domains. Further it creates missing directories for the configuration and prints warnings pertaining to problems in the configuration.

The command **--check-key** (or just **--check**) checks whether a key with the given user-id is installed. The process returns success in this case; to also print a diagnostic use the option **-v**. If the key is not installed a diagnostic is printed and the process returns failure; to suppress the diagnostic, use option **-q**. More than one user-id can be given; see also option **with-file**.

The command **--install-key** manually installs a key into the WKD. The arguments are a file with the keyblock and the user-id to install. If the first argument resembles a fingerprint the key is taken from the current keyring; to force the use of a file, prefix the first argument with "./". If no arguments are given the parameters are read from stdin; the expected format are lines with the fingerprint and the mailbox separated by a space.

The command **--remove-key** uninstalls a key from the WKD. The process returns success in this case;

to also print a diagnostic, use option **-v**. If the key is not installed a diagnostic is printed and the process returns failure; to suppress the diagnostic, use option **-q**.

The command **--revoke-key** is not yet functional.

OPTIONS

gpg-wks-server understands these options:

-C *dir*
--directory *dir*

Use *dir* as top level directory for domains. The default is '*/var/lib/gnupg/wks*'.

--from *mailaddr*

Use *mailaddr* as the default sender address.

--header *name*=*value*

Add the mail header "*name*: *value*" to all outgoing mails.

--send

Directly send created mails using the **sendmail** command. Requires installation of that command.

-o *file*
--output *file*

Write the created mail also to *file*. Note that the value - for *file* would write it to stdout.

--with-dir

When used with the command **--list-domains** print for each installed domain the domain name and its directory name.

--with-file

When used with the command **--check-key** print for each user-id, the address, ’i’ for installed key or ’n’ for not installed key, and the filename.

--verbose
Enable extra informational output.

--quiet
Disable almost all informational output.

--version
Print version of the program and exit.

--help
Display a brief help page and exit.

EXAMPLES

The Web Key Service requires a working directory to store keys pending for publication. As root create a working directory:

```
# mkdir /var/lib/gnupg/wks
# chown webkey:webkey /var/lib/gnupg/wks
# chmod 2750 /var/lib/gnupg/wks
```

Then under your webkey account create directories for all your domains. Here we do it for "example.net":

```
$ mkdir /var/lib/gnupg/wks/example.net
```

Finally run

```
$ gpg-wks-server --list-domains
```

to create the required sub-directories with the permissions set correctly. For each domain a submission address needs to be configured. All service mails are directed to that address. It can be the same address for all configured domains, for example:

```
$ cd /var/lib/gnupg/wks/example.net
$ echo key-submission@example.net >submission-address
```

The protocol requires that the key to be published is sent with an encrypted mail to the service. Thus you need to create a key for the submission address:

```
$ gpg --batch --passphrase ’’ --quick-gen-key key-submission@example.net
$ gpg -K key-submission@example.net
```

The output of the last command looks similar to this:

```
sec   rsa3072 2016-08-30 [SC]
      C0FCF8642D830C53246211400346653590B3795B
uid           [ultimate] key-submission@example.net
              bxzcxpxk8h87z1k7bzk86xn5aj47intu@example.net
ssb   rsa3072 2016-08-30 [E]
```

Take the fingerprint from that output and manually publish the key:

```
$ gpg-wks-server --install-key C0FCF8642D830C53246211400346653590B3795B \
>                key-submission@example.net
```

Finally that submission address needs to be redirected to a script running **gpg-wks-server**. The **procmail** command can be used for this: Redirect the submission address to the user "webkey" and put this into webkey’s ‘*.procmailrc*’:

```
:0
* !^From: webkey@example.net
* !^X-WKS-Loop: webkey.example.net
|gpg-wks-server -v --receive \
    --header X-WKS-Loop=webkey.example.net \
    --from webkey@example.net --send
```

SEE ALSO

gpg-wks-client(1)