What is a Web Key Directory?

Web Key Directories provide an easy way to discover public keys through HTTPS. They provide an important piece to the infrastructure **to improve the user experience for exchanging secure emails and files**.

In contrast to the public keyservers a Web Key Directory does not publish mail addresses. And it is an authoritative pubkey source for its domain.

How does it work?

1. The sender's mail client checks a "well known" URL on the domain of the recipient.
2. If a public key is available for that mail address, will be downloaded via HTTPS.
3. The downloaded pubkey can now be used without further user interaction.

Such an URL looks like: https://intevation.de/.well-known/openpgpkey/hu/g8td9rsyatrazsoiho37j9n3g5ypp34h for the mail address "aheinecke@intevation.de"

What does it mean for users?

A user just selects the recipients of a message and by default the encryption state of that mail will toggle if all recipients can be found in a Web Key Directory.

[Example from Gpg4win / GpgOL](https://files.intevation.de/users/aheinecke/wkd-autoencrypt.gif)

For a basic level of security the user does **not need to check a fingerprint** or do any key management manually.

How to set it up?

If you want to set up a Web Key Directory for your own server or your own server you only need access to a webserver for your domain. See: [WKDHosting](https://wiki.gnupg.org/WKDHosting)

For a larger organization it is recommended to set up a complete Web Key Service, which will help to automate Web Key Directory publishing.

Stopgap method - temporary central keyserver

Not recommended - but a temporary workaround - is to use "WKDaaS" and delegate delivery of your pubkey to a central service. Doing this you'll expose all people that want to use crypto when communication with you towards another party of the central service or monitoring the central service. This third party can then see the communication pattern.

However this maybe a temporary solution until you will convince your mail provider to enable at least the WKD serving part or to switch to a more privacy aware mail provider.

One service is keys.openpgp.org, where you can set the CNAME record of the "openpgpkey" subdomainto "wkd.keys.openpgp.org" the [CNAME](https://wiki.gnupg.org/CNAME) entry should look like this.

openpgpkey.example.org. 300 IN CNAME wkd.keys.openpgp.org.

In addition you need to register your pubkey with them.

Other drawbacks:

* As any WKD service, they'll be able to serve a different pubkey to some domains at some time, however opposed to your email provider you do not have a contractual relationship with them.
* Elder GnuPGs like the some on Debian Stretch do not offer the necessary modern WKD implementation for a successful request, so you are reaching less communication partners with this compared to real WKD.
* (For the overall ecosystem, we need more decentral services instead, it is at the core of OpenPGP security promise. So you are missing to set a good example. ;) )

Web Key Directory (WKD) / Web Key Service (WKS) what is the difference?

The Web Key Directory is the HTTPS directory from which keys can be fetched.

The Web Key Service is a tool / protocol to automatically publish and update keys in the Web Key Directory. It is **optional** to reduce the administrative effort of a Web Key Directory.

Documentation how to set up a Web Key Service can be found on the [Web Key Service page](https://wiki.gnupg.org/WKS).

Technical Details

You can find the concepts / technical details under [WKDDetails](https://wiki.gnupg.org/WKDDetails).

Trust and security considerations are outlined as part of the [AutomatedEncryption](https://wiki.gnupg.org/AutomatedEncryption) concept.

Implementations

GnuPG

* WKD lookup is implemented in [GnuPG](https://wiki.gnupg.org/GnuPG) since v2.1.12. It is enabled by default since 2.1.23..
* WKS server and client tools are part of [GnuPG](https://wiki.gnupg.org/GnuPG) since v2.1.14

Mail Clients

Any mail client which uses the --locate-keys option of [GnuPG](https://wiki.gnupg.org/GnuPG) will automatically do WKD requests.

Known mail clients with WKD Support:

* Thunderbird/[Enigmail 2.0](https://www.enigmail.net/index.php/en/download/changelog)
* KMail since Version 5.6
* Outlook with [GpgOL](https://wiki.gnupg.org/GpgOL) since Version 2.2.0
* Mailvelope since Version 3.0.0 (Dez 2018)
* [K9Mail](https://wiki.gnupg.org/K9Mail) with [OpenKeyChain](https://wiki.gnupg.org/OpenKeyChain) since Version 5.1 (Jun 2018)

Known mail clients with WKS Support:

* Thunderbird/[Enigmail 2.0](https://www.enigmail.net/index.php/en/download/changelog)
* KMail since Version 5.6
* Outlook with [GpgOL](https://wiki.gnupg.org/GpgOL) (basic, pre-release) since Version 2.2.1 asyGpg2016)

Self-hosted email setups offering WKD + WKS:

* [caesonia - OpenBSD email service](https://github.com/vedetta-com/caesonia/).
* [Excision Mail - OpenBSD email service using ansible](https://github.com/Excision-Mail/Excision-Mail/): Has multiple-domain WKS support.

Mail Service Providers offering WKD

* [Posteo](https://posteo.de/en/) offers web key directory lookup and service for @posteo.de-addresses (**Since 2016-12**) E.g. [German Thunderbird/WKD Instructions](https://posteo.de/hilfe/easygpg-wie-veroeffentliche-ich-meinen-oeffentlichen-pgp-schluessel-ueber-web-key-directory-wkd-im-posteo-schluesselverzeichnis)
* [Protonmail](https://protonmail.com/) supports web key directory lookup (**Since 2018-11**)
* [netzguerilla](https://netzguerilla.net/) offers web key directory lookup. (**Since 2017-10-11**)
* [mailbox.org](https://mailbox.org/en/) **plans** to offer web key directory lookup in Q2 2018 (coming with [OX Guard 2.10](https://knowledgebase.open-xchange.com/roadmap.html#21)).

Organizations using WKD

* [C3S](https://www.c3s.cc/)
* [cotech.de](https://www.cotech.de/)
* [debian.org](https://www.debian.org/)
* [gentoo.org](https://gentoo.org/)
* [gnupg.org](https://gnupg.org/) (Testing accounts available for developers implementing WKD in MUAs.)
* [KDAB.com](https://kdab.com/)
* [kernel.org](https://kernel.org/)
* [occrp.org](https://www.occrp.org/)
* [torproject.org](https://www.torproject.org/)
* [f-droid.org](https://f-droid.org/)
* [guardianproject.info](https://guardianproject.info/)
* [privacytools.io](https://privacytools.io/)
* (Several smaller organizations. Like - unsurprisingly - g10code.com and intevation.de. *Let us know if you want to be publicly listed.*)

**Setting up OpenPGP Web Key Directory (WKD)**

If you use OpenPGP to secure your email communication, you should consider publishing your public key using Web Key Directory. It's easier than you think.

* 

**Freddie Leeman**

Geek, dad, entrepreneur, security enthusiast, thrill seeker.

[More posts](https://www.uriports.com/blog/author/freddie/) by Freddie Leeman.

[](https://www.uriports.com/blog/author/freddie/)

[FREDDIE LEEMAN](https://www.uriports.com/blog/author/freddie/)

12 SEP 2019 • 3 MIN READ



If you value privacy and use OpenPGP to secure your email communication and have a secure web server running on the same domain, you might consider publishing your public key through [Web Key Directory](https://www.ietf.org/id/draft-koch-openpgp-webkey-service.txt). This allows supporting email clients to automatically retrieve your public key, using the HTTPS protocol.

**How does it work?**

When a sender uses a supporting mail client and adds an email address to a message, the client will automatically check if WKD is set up for the domain name of the receiver. If a public key is found, it is directly imported to the keyring allowing secure communication.

**Advanced vs Direct**

There are two ways to implement WKD. The first is the **advanced**method, which is a bit harder to set up and requires a CA signed and trusted certificate for the "*openpgpkey*" sub-domain*.*The **direct** method requires no additional DNS entries but you have to make sure that the "*openpgpkey"* sub-domain does not exist and is not subject to wildcarding. If you use a wildcard for the domain you need to insert an empty TXT RR for the "*openpgpkey"* sub-domain.

For the **advanced**implementation, create the following folder inside your web root folder for sub-domain "*openpgpkey.example.org"*:

/.well-known/openpgpkey/example.org/hu/

If you're going to implement the **direct**method, create the following folder inside your web root folder:

/.well-known/openpgpkey/hu/

After you've created the folder, add an empty policy file to let clients know that you've set up the WKD service. Put the file in the "*/hu/*"-**parent** folder and check if you can access the empty file with your browser.

**Advanced:**  
https://openpgpkey.example.org/.well-known/openpgpkey/example.org/policy

**Direct:**  
https://example.org/.well-known/openpgpkey/policy

If the link above works we need to make a few adjustments to the web server setup before adding the public keys to the "*/hu/"* folder.

**Nginx**

location /.well-known/openpgpkey/hu/ {

default\_type "application/octet-stream";

add\_header Access-Control-Allow-Origin \* always;

}

**Apache (config)**

<Directory "/.well-known/openpgpkey/hu">

<IfModule mod\_mime.c>

ForceType application/octet-stream

</IfModule>

<IfModule mod\_headers.c>

Header always set Access-Control-Allow-Origin "\*"

</IfModule>

</Directory>

**Apache (.htaccess in /hu/ folder)**

<IfModule mod\_mime.c>

ForceType application/octet-stream

</IfModule>

<IfModule mod\_headers.c>

Header always set Access-Control-Allow-Origin "\*"

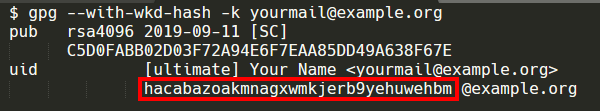
</IfModule>

All that's left to do now is figure out what the hash is of the local-part of your email address and save a binary (not ASCII armored) version of your public key under that name in the "*/hu/*" folder.

**Local-part hash as filename**

The public key lookup is done using a calculated fixed-sized string of the local-part of the email address (the part before the @-symbol). By hashing it using a SHA-1 algorithm and then encoding the result using a Z-Base-32 method, you are left with 32 octets that should be used as the filename for your key. Calculating this string sounds complicated but fortunately we can use the ***gpg*** (GnuPG v2.1.12 or newer) command below :

gpg --with-wkd-hash -k yourmail@example.org



The 32 characters outlined in the image above is the filename we want to save the public key to using the following command:

gpg --export yourmail@example.org > hacabazoakmnagxwmkjerb9yehuwehbm

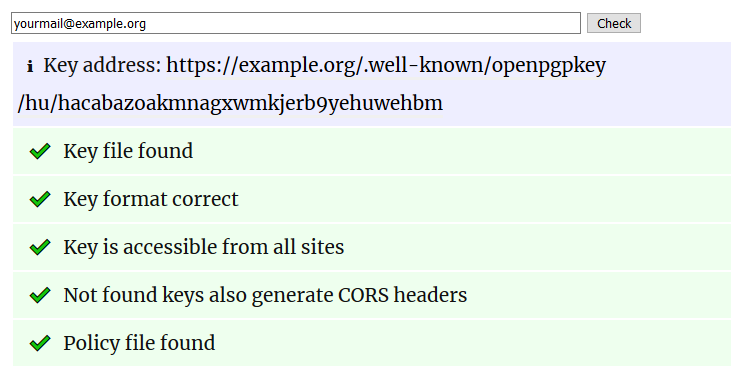
Move the created file to the "*/hu/"* folder and check if the file is downloadable using the links below:

**Advanced:**  
https://openpgpkey.example.org/.well-known/openpgpkey/example.org/hu/hacabazoakmnagxwmkjerb9yehuwehbm

**Direct:**  
https://example.org/.well-known/openpgpkey/policy/hu/hacabazoakmnagxwmkjerb9yehuwehbm

**Done!**

Once the file is saved and accessible you are all set! Test if the configuration works by using a WKD validator like this one: <https://metacode.biz/openpgp/web-key-directory>

Metacode WKD check results example

Congratulations! As [more and more](https://wiki.gnupg.org/WKD#Implementations) mail clients and mail service providers are adopting WKD, you'll automatically receive more encrypted messages from new communication partners without off-channel key exchange or the need for them to search the network of public key servers.

**Email security**

While you are taking your email configuration to the next level, have you configured MTA-STS, DANE, SPF, DKIM and DMARC yet? More on the subject of email security can be found in my blog [here](https://www.uriports.com/blog/email-security-explained/).