NAME

porthash − compute, or verify (default), signed hash of Portage repo

SYNOPSIS

porthash [*options*]

DESCRIPTION

porthash is a simple script to save, or by default verify, a **signed master hash** of the specified Portage repository tree (by default, */usr/portage*). It is intended to provide assurance - when distributing a Portage repo snapshot (whether of main **gentoo repo**, or a custom overlay) over an unauthenticated channel (e.g. **rsync**(1)) - that the consitutent ebuilds, manifests etc. have not been tampered with in transit.

The master hash is calculated by first creating a sorted list of hashes of the contents of each file in the repo tree (excluding *distfiles*, *packages* and *.git* directories, optionally the *metadata* directory, and the hash (*repo.hash*) and signature (*repo.hash.asc*) files), then hashing that list itself and temporarily storing the result.

Next, a sorted list of hashes of the **stat**(1) of all files *and* directories in the repo tree is calculated (with the same excluded set), and *that* resulting list hashed and temporarily stored. (The hashed **stat**(1) output consists of the relative file path, octal permissions, file type, owner name and group name for each file and (sub)directory.)

Then, the two stored intermediate hashes are concatenated and hashed again, to create the final result (the "master hash").

When **porthash** is invoked *with* the **--create** option, the master hash is saved (along with some metadata) in the file *repo.hash* in the top-level repo directory. It is then signed (by default, using private key **5D90CAF4** (which must exist in root's default keyring and be usable without passphrase) unless a different one is specified via the **--key** option), and the signature stored (in ascii form) in *repo.hash.asc*, also in the top-level repo directory.

When invoked *without* the **--create** option (default), the validity of *repo.hash* is first checked, and its signature verified (by default, against the public key **5D90CAF4** (which must exist in root's default keyring) unless a different one is specified via the **--key** option). If OK, then the computed master hash is compared with that stored in *repo.hash*, and, if they match, the repo is declared valid; otherwise an error is reported (and a non-zero error code returned, see **EXIT STATUS** below).

This version of **porthash** uses **sha512sum**(1) as its hash function.

OPTIONS

−c, **−−create**

Instead of validating an existing hash (the default), computes a new master hash, saves to *repo.hash*, signs it and saves the signature to *repo.hash.asc*.

Any previous *repo.hash* and *repo.hash.asc* for the repo will be overwritten by this process.

−h, **−−help**

Displays a short help screen, and exits.

−−key=KEYID

Use the specified KEYID to select the appropriate **gpg2**(1) public key (when verifying) or private key (when creating) the hash signature. Defaults to **5D90CAF4** if not specified.

−m, **−−metadata**

Do not omit the *metadata* directory when creating a new hash (it is omitted by default). This option is ignored when **verifying** a hash, since the *repo.hash* file contains a flag specifying

whether or not this directory was considered at the time of hash creation.

NB: if you do use the **-m** flag, take care that e.g. regenerated md5-cache entries (or similar) do not end up causing false hash validation failures at the client end.

−−repo=RDIR
Specify the base directory for the repo in question. Defaults to */usr/portage* if unspecified.

−v, **−−version**
Displays the version number of **porthash**, and exits.

EXIT STATUS

The following exit status codes may be returned by *porthash* when verifying an existing signed master hash:

- **0** no error
- **1** misc. error (bad option etc.)
- **2** the file *repo.hash* does not exist
- **3** the file *repo.hash.asc* does not exist
- **4** *repo.hash.asc* not a valid signature for *repo.hash*
- **5** *repo.hash.asc* issued by the wrong signer
- **6** *repo.hash* has an unknown format (too modern?)
- **7** *repo.hash* does not match our computed master hash
- **11** specified public key not in root's default keyring

And following exit status codes may be returned by *porthash* when creating a new hash (**--create** option):

- **0** no error
- **1** misc. error (bad option etc.)
- **8** failed to create *repo.hash*
- **9** failed to create *repo.hash.asc*
- **10** specified private key not in root's default keyring

BUGS

Although **porthash** does ignore the *.git* directory if present, it does **not** consider the contents of any *.gitignore* file. This is not usually a problem, as long as (where the repo in question is checked in under **git**(1)), the master hash is computed in a clone of the checked-in tree, rather than in the working tree directly.

COPYRIGHT

AUTHORS

sakaki — send bug reports or comments to <sakaki@deciban.com>

SEE ALSO

git(1), **gpg2**(1), **rsync**(1), **sha512sum**(1), **stat**(1), **portage**(5).