NAME

pkg key - signing key operations

SYNOPSIS

pkg key [**--create** | **--public** | **--sign**] [**-t** *type*] *keyfile*

DESCRIPTION

pkg key is used to create or extract signing keys for use with pkg-repo(8). Cryptographically signing your package repository catalog is strongly recommended.

One of the **--create**, **--public**, or **--sign** operations must be specified. Future work may write information about the *keyfile* out to *stdout* when no key operation has been specified.

See pkg-repo for some practical examples of using **pkg key**.

OPTIONS

The following options are supported by **pkg key**:

--create

Create the named key. Note that any file at *keyfile* will be overwritten. **pkg key** will chmod(2) the *keyfile* to 0400 upon successful completion. The corresponding public key will be written to *stdout*, note the caveats of this described with the **--public** option. The **-t** option should be used when generating keys to be explicit about the type of key requested.

Note that the **ecdsa** and **eddsa** keys generated by **pkg key** are not compatible with those generated by OpenSSL, but pkg(8) can read **ecdsa** keys generated by OpenSSL.

--public

Write the public key corresponding to *keyfile* out to *stdout*. Note that some signers may output keys in a binary format, so it is recommended to redirect *stdout* to a file.

--sign

Signs the data ingested via *stdin* with the named *keyfile*, and writes the signature data to *stdout*. As with **--public**, note that the signature may be a binary format and it is recommended to redirect *stdout* to a file.

-t *type*

Specifies the *type* of signer to use for the given key. **pkg key** will not try to guess the correct signer that goes with a key in any case, so it must be specified for every operation. The **rsa** signer is assumed if **-t** is not specified. The following signers are currently supported:

rsa
Backend using RSA with keys created either by OpenSSL or by **pkg key --create**.

ecc
An alias for the **eddsa** signer.

ecdsa
Backend using ECDSA with keys created either by OpenSSL or by **pkg key --create**. See *Elliptic Curve Cryptography* for more discussion.

eddsa
Backend using EdDSA with keys created by **pkg key --create**. See *Elliptic Curve Cryptography* for more discussion.

Elliptic Curve Cryptography

Elliptic Curve Cryptography, ECC, is supported by pkg(8), with limited compatibility with OpenSSL. Signatures are output in a format that OpenSSL can handle, subject to the constraints about curve choice outlined in the rest of this section.

The **ecdsa** signer is expected to be interoperable with OpenSSL, but curve choice is more limited than what OpenSSL provides. In general, the curves provided must be supported both by OpenSSL and by the library "libecc" used by pkg(8). The criteria for curve selection is that they must be 256-bit or higher and accepted by both implementations. The following common curves are currently supported:

- **secp256k1**
- **secp384r1**
- **secp521r1**
- **brainpoolP256r1**
- **brainpoolP256t1**
- **brainpoolP320r1**
- **brainpoolP320t1**
- **brainpoolP384r1**

- **brainpoolP384t1**
- **brainpoolP512r1**
- **brainpoolP512t1**

The **eddsa** signer is not compatible with OpenSSL due to limited curve selection provided by library "libecc" by default. The only curve supported by pkg(8) for EdDSA is **WEI25519**.

FILES

See pkg.conf(5).

SEE ALSO

pkg-repo(8)