NAME

aed — perform aes256-cbc encryption/decryption

SYNOPSIS

aed [**−deh**]

DESCRIPTION

The **aed** utility can be used to perform symmetric encryption/decryption of the input stream using 256bit AES with a SHA256 digest.

OPTIONS

aed supports the following command-line options:

−d Perform decryption of the input stream.

−e Perform encryption of the input stream.

−h Print a short usage and exit.

DETAILS

aed reads data from stdin and either encrypts or decrypts it (depending on the **−d** or **−e** flag). It uses AES 256bit CBC mode with a SHA256 digest with keying material derived from the passphrase using the `EVP_BytesToKey`(3) function, generating a suitable salt via `RAND_bytes`(3).

aed reads the password from which to derive the key material from the *AED_PASS* environment variable.

When encrypting, the output is prefixed by the string "Salted__", followed by the 8 byte salt. Output is written to stdout.

EXAMPLES

To encrypt the contents of the file ’file’ and storing the encrypted output in ’file.enc’:

```
aed -e <file >file.enc
```

To decrypt the contents of that file again:

```
aed -d <file.enc
```

Since **aed** operates on stdin and stdout, the above two commands could also be chained:

```
cat file | aed -e | aed -d
```

EXIT STATUS

aed exits 0 on success, and >0 if an error occurred.

SEE ALSO

`EVP_BytesToKey`(3), `EVP_EncryptInit`(3), `RAND_bytes`

https://www.netmeister.org/blog/passing-passwords.html

HISTORY

This program (or variants thereof) was first assigned as a stand-alone programming assignment for the class “Advanced Programming in the UNIX Environment” at Stevens Institute of Technology in the Fall of 2012.

BUGS

Well, let’s see...