NAME

openssl-rsa, rsa - RSA key processing tool

SYNOPSIS

openssl rsa [**-help**] [**-inform PEM|NET|DER**] [**-outform PEM|NET|DER**] [**-in filename**] [**-passin arg**] [**-out filename**] [**-passout arg**] [**-aes128**] [**-aes192**] [**-aes256**] [**-camellia128**] [**-camellia192**] [**-camellia256**] [**-des**] [**-des3**] [**-idea**] [**-text**] [**-noout**] [**-modulus**] [**-check**] [**-pubin**] [**-pubout**] [**-RSAPublicKey_in**] [**-RSAPublicKey_out**] [**-engine id**]

DESCRIPTION

The **rsa** command processes RSA keys. They can be converted between various forms and their components printed out. **Note** this command uses the traditional SSLeay compatible format for private key encryption: newer applications should use the more secure PKCS#8 format using the **pkcs8** utility.

OPTIONS

-help
Print out a usage message.

-inform DER|NET|PEM
This specifies the input format. The **DER** option uses an ASN1 DER encoded form compatible with the PKCS#1 RSAPrivateKey or SubjectPublicKeyInfo format. The **PEM** form is the default format: it consists of the **DER** format base64 encoded with additional header and footer lines. On input PKCS#8 format private keys are also accepted. The **NET** form is a format is described in the **NOTES** section.

-outform DER|NET|PEM
This specifies the output format, the options have the same meaning as the **-inform** option.

-in filename
This specifies the input filename to read a key from or standard input if this option is not specified. If the key is encrypted a pass phrase will be prompted for.

-passin arg
the input file password source. For more information about the format of **arg** see the **PASS PHRASE ARGUMENTS** section in *openssl(1)* .

-out filename
This specifies the output filename to write a key to or standard output if this option is not specified. If any encryption options are set then a pass phrase will be prompted for. The output filename should **not** be the same as the input filename.

-passout password
the output file password source. For more information about the format of **arg** see the **PASS PHRASE ARGUMENTS** section in *openssl(1)* .

-aes128|-aes192|-aes256|-camellia128|-camellia192|-camellia256|-des|-des3|-idea
These options encrypt the private key with the specified cipher before outputting it. A pass phrase is prompted for. If none of these options is specified the key is written in plain text. This means that using the **rsa** utility to read in an encrypted key with no encryption option can be used to remove the pass phrase from a key, or by setting the encryption options it can be use to add or change the pass phrase. These options can only be used with PEM format output files.

-text
prints out the various public or private key components in plain text in addition to the encoded version.

-noout
this option prevents output of the encoded version of the key.

-modulus
this option prints out the value of the modulus of the key.

-check
this option checks the consistency of an RSA private key.

-pubin
by default a private key is read from the input file: with this option a public key is read instead.

-pubout
by default a private key is output: with this option a public key will be output instead. This option is automatically set if the input is a public key.

-RSAPublicKey_in, **-RSAPublicKey_out**
like **-pubin** and **-pubout** except **RSAPublicKey** format is used instead.

-engine id
specifying an engine (by its unique **id** string) will cause **rsa** to attempt to obtain a functional reference to the specified engine, thus initialising it if needed. The engine will then be set as the default for all available algorithms.

NOTES

The PEM private key format uses the header and footer lines:

```
-----BEGIN RSA PRIVATE KEY-----
-----END RSA PRIVATE KEY-----
```

The PEM public key format uses the header and footer lines:

```
-----BEGIN PUBLIC KEY-----
-----END PUBLIC KEY-----
```

The PEM **RSAPublicKey** format uses the header and footer lines:

```
-----BEGIN RSA PUBLIC KEY-----
-----END RSA PUBLIC KEY-----
```

The **NET** form is a format compatible with older Netscape servers and Microsoft IIS .key files, this uses unsalted RC4 for its encryption. It is not very secure and so should only be used when necessary.

Some newer version of IIS have additional data in the exported .key files. To use these with the utility, view the file with a binary editor and look for the string ‘‘private-key’’, then trace back to the byte sequence 0x30, 0x82 (this is an ASN1 SEQUENCE). Copy all the data from this point onwards to another file and use that as the input to the **rsa** utility with the **-inform NET** option.

EXAMPLES

To remove the pass phrase on an RSA private key:

```
openssl rsa -in key.pem -out keyout.pem
```

To encrypt a private key using triple DES:

```
openssl rsa -in key.pem -des3 -out keyout.pem
```

To convert a private key from PEM to DER format:

```
openssl rsa -in key.pem -outform DER -out keyout.der
```

To print out the components of a private key to standard output:

```
openssl rsa -in key.pem -text -noout
```

To just output the public part of a private key:

```
openssl rsa -in key.pem -pubout -out pubkey.pem
```

Output the public part of a private key in **RSAPublicKey** format:

```
openssl rsa -in key.pem -RSAPublicKey_out -out pubkey.pem
```

BUGS

The command line password arguments don’t currently work with **NET** format.

There should be an option that automatically handles .key files, without having to manually edit them.

SEE ALSO

pkcs8(1) , *dsa(1)* , *genrsa(1)* , *gendsa(1)*

COPYRIGHT

Copyright 2000-2016 The OpenSSL Project Authors. All Rights Reserved.

Licensed under the OpenSSL license (the ‘‘License’’). You may not use this file except in compliance with the License. You can obtain a copy in the file LICENSE in the source distribution or at <https://www.openssl.org/source/license.html>.