- Aluno: Quenio Cesar Machado dos Santos
- Matricula: 14100868

- Disciplina: INE5429 - Segurança em Computadores
- Data: 19/04/2016

openssl

ans1parse

Este comando permite gerar e extrair dados formatados em ASN.1.

Exemplo:

```
$ openssl asn1parse -genstr 'UTF8:Hello World'
    0:d=0  hl=2 l=  11 prim: UTF8STRING        :Hello World
```

ciphers

Lista os algoritmos de criptografia disponíveis.

Exemplo:

```
$ openssl ciphers
DHE-RSA-AES256-SHA:DHE-DSS-AES256-SHA:AES256-SHA:EDH-RSA-DES-CBC3-SHA:EDH-DSS-DES-CBC3-SHA:DES-CBC3-SHA:DES-CBC3-MD5:DHE-RSA-AES128-SHA:DHE-DS
```

dgst

Gera o "hash" de um arquivo usando o algoritmo especificado.

A opção "-md5" usa o algoritmo MD5.

A opção "-hex" gera o "hash" em hexadecimal.

Exemplo:

```
$ openssl dgst -md5 -hex quenio.gpg.txt
MD5(quenio.gpg.txt)= f2bcfa699e2800ad29e8db3f2dcd5079
```

enc

Permite cifrar ou decifrar arquivos usando vários tipos de algoritmos simétricos.

A opção "-salt" é usada quando a chave é gerada de uma senha, pois impede ataque por dicionários de senha.

Exemplo:

```
$ openssl des3 -salt -in ./quenio.gpg.txt -out ./quenio.gpg.des3
enter des-ede3-cbc encryption password:
Verifying - enter des-ede3-cbc encryption password:

$ ls -l quenio.gpg.des3
-rw-r--r--  1 Quenio  staff  1784 19 Abr 17:15 quenio.gpg.des3
```

passwd

Gera o "hash" de uma senha.

Exemplo:

```
openssl passwd -crypt -salt xx password prints xxj31ZMTZzkVA
xxj31ZMTZzkVA
xxkNUZCRnGcOs
Warning: truncating password to 8 characters
xxRNLt2njILYc
```

rand

Gera dados pseudo-randomicos a partir de um número-semente.

Exemplo:

```
$ openssl rand -base64 345
ACbHmnAK6CrrUPUAz28r0STnolo41cPsAJUSakRqvTOdMak0EqGeeQCedfd32z21
iA0PzZdDDt20Yl70HqBNOI7vvLpgYDKWA0JHwM6nxE+ajDSk6eWgyMC6xCx9dpKu
h9sKQvFWaLMVJ46RkA5XTDj70WCKO41lbSk/Io59ySzDh1sbQuEstA+91VYNwrC5
gM4W7NV7yLSqDsUB06nq9RDUWWrPL88nJ0/Bjzip21NgbuNMIPt+QfOWEFSUweuQ
ujb4cmTB4eVP9GXBxcFBNC6dP8+TYdenNYcuy9h+cA0Tqkyr+lmVSIk2pmJZxv9C
xHjc0ZNX2dW0sZbC8yy1SbzoRsBBXAl7R27+eKqKUo0c9iVL4PQgUL5I4US0845q
07sey9yGBzQ+bOiB1q+T/8CRyQkREzFRs1at0P9d7I+7un5suGwQ3EaQVvFdUsDG
3/8qoQOzdfQs
```

```
$ openssl rand -base64 345
Z3x2lUnCqBItpYDlIWuzJFKzWlpDYd7DOELhq4a1jxwZztaAb+haEuzSz0Ifr3LP
dIgOd1dLIqaA6Z5e4fDzbx9AK1g9CrcEhWf5xM5pW6hC/hKMrt5zCNeHFeyCfwvu
Ws5BRf+m6RlCf3/bHq2ROGjBhpBgql82Sd3DhiPo8bIxgGGIO7TwSlcwwQKjrsF0
4PmkR6xmTZZLdIrIkBq2plgkVsjix2mMp8SseCDIA407+G9vPEVl5wp6F6fS/N0R
WPQFHSw0PRx6d4GMZXDqqIvcewrxJAvqx6hLlRN3W1u9DBXqK31tC5pQsujjFG6O
OtoD7w+Sen+/SndNitMf/YyImCq+RO4nsVOLBpnE9E/VowXJUfXzOCcHg/eqrsyo
ixRtkpslWUeAlyR/qF3i8CptHCiEdI69bY8Nk2RzuQ6PZRklv4FphHp+9JbV4q+l
bNb5z4iuXr5l
```

speed

Usado para testar o desempenho dos vários algoritmos de criptografia.

Exemplo:

```
$ openssl speed aes-256-cbc
To get the most accurate results, try to run this
program when this computer is idle.
Doing aes-256 cbc for 3s on 16 size blocks: 19706632 aes-256 cbc's in 3.00s
Doing aes-256 cbc for 3s on 64 size blocks: 5157101 aes-256 cbc's in 3.00s
Doing aes-256 cbc for 3s on 256 size blocks: 1284706 aes-256 cbc's in 2.99s
Doing aes-256 cbc for 3s on 1024 size blocks: 328193 aes-256 cbc's in 2.99s
Doing aes-256 cbc for 3s on 8192 size blocks: 40993 aes-256 cbc's in 3.00s
OpenSSL 0.9.8zh 14 Jan 2016
built on: Feb  4 2016
options:bn(64,64) md2(int) rc4(ptr,char) des(idx,cisc,16,int) aes(partial) blowfish(idx)
compiler: -arch x86_64 -fmessage-length=0 -pipe -Wno-trigraphs -fpascal-strings -fasm-blocks -O3 -D_REENTRANT -DDSO_DLFCN -DHAVE_DLFCN_H -DL_EI
available timing options: TIMEB USE_TOD HZ=100 [sysconf value]
timing function used: getrusage
The 'numbers' are in 1000s of bytes per second processed.
type             16 bytes     64 bytes    256 bytes   1024 bytes   8192 bytes
aes-256 cbc     105249.81k   110157.17k   109813.17k   112239.53k   112018.46k
```