Rocky Enterprise Linux 9.2 Manual Pages on command 'apg.1'

C:\>man apg.1

APG(1) User Manual APG(1)

NAME

apg - generates several random passwords

SYNOPSIS

apg [-a algorithm] [-M mode] [-E char_string] [-n num_of_pass] [-m min_pass_len] [-x max_pass_len] [-r dictfile] [-b filter_file] [-p min_substr_len] [-s] [-c cl_seed] [-d] [-y] [-l] [-t] [-q] [-h] [-v]

DESCRIPTION

apg generates several random passwords. It uses several password generation algorithms (currently two) and a built-in pseudo random number generator.

Default algorithm is pronounceable password generation algorithm designed by Morrie Gasser and described in A Random Word Generator For Pronounceable Passwords National Technical Information Service (NTIS) AD-A-017676. The original paper is very old and had never been put online, so I have to use NIST implementation described in FIPS-181.

Another algorithm is simple random character generation algorithm, but it uses four user-defined symbol sets to produce random password. It means that user can choose type of symbols that should appear in password. Symbol sets are: numeric symbol set (0,...,9), capital letters symbol set (A,...,Z), small letters symbol set (a,...,z) and special symbols symbol set (#,@,!,...).

Built-in pseudo random number generator is an implementation of algorithm described in Appendix C of ANSI X9.17 or RFC 1750 with exception that it uses CAST or SHA-1

instead of Triple DES. It uses local time with precision of microseconds (see get? timeofday(2)) and /dev/random (if available) to produce initial random seed.

apg also have the ability to check generated password quality using dictionary. You can use this ability if you specify command-line options -r dictfile or -b filter? name where dictfile is the dictionary file name and filtername is the name of Bloom filter file. In that dictionary you may place words (one per line) that should not appear as generated passwords. For example: user names, common words, etc. You even can use one of the dictionaries that come with dictionary password crackers. Bloom filter file should be created with apgbfm(1) utility included in apg distribution.

In future releases I plan to implement some other techniques to check passwords (like pattern check) just to make life easier.

OPTIONS

Password generation modes options

-a algorithm

Use algorithm for password generation.

0 - pronounceable password generation (default)

1 - random character password generation

-n num_of_pass

Generate num_of_pass number of passwords. Default is 6.

-m min_pass_len

Generate password with minimum length min_pass_len. If min_pass_len > max_pass_len then max_pass_len = min_pass_len. Default minimum password length is 8.

-x max_pass_len

Generate password with maximum length max_pass_len. If min_pass_len > max_pass_len then max_pass_len = min_pass_len. Default maximum password length is 10.

-M mode

Use symbolsets specified with mode for password generation. mode is a text string consisting of characters S, s, N, n, C, c, L, l. Where:

S Generator must use special symbol set for every generated password.

s Generator should use special symbol set for password generation.

N Generator must use numeral symbol set for every generated password.

n Generator should use numeral symbol set for password generation.

C Generator must use capital symbol set for every generated password.

c Generator should use capital symbol set for password generation.

L Generator must use small letters symbol set for every generated pass?
word (always present if pronounceable password generation algorithm
is used).

l Generator should use small letters symbol set for password genera?
tion.

R,r Not supported any more. Use -E char_string option instead.

mode can not be more than 4 characters in length.

Note:

Usage of L, M, S, C will slow down password generation process.

Examples:

-M sncl

-M SNCL

-M Cn

-E char_string

Exclude characters in char_string from password generation process (in pro?
nounceable password generation mode you can not exclude small letters). To
include special symbols that can be recognized by shell (apostrophe, quotes,
dollar sign, etc.) in char_string use the backslashed versions.

Examples:

Command apg -a 1 -M n -n 3 -m 8 -E 23456789 will generate a set of passwords
that will look like this:

10100110

01111000

11011101

Command apg -a 1 -M nc -n 3 -m 26 -E GHIJKLMNOPQRSTUVWXYZ will generate a
set of passwords that will look like this:

16A1653CD4DE5E7BD9584A3476

C8F78E06944AFD57FB9CB882BC

8C8DF37CD792D36D056BBD5002

-r dictfile

Check generated passwords for their appearance in dictfile

-b filter_file

Check generated passwords for their appearance in filter_file. filter_file should be created with the apgbfm(1) utility.

-p min_substr_len

This option tells apg(1) to check every substring of the generated password for appearance in filter_file. If any of such substrings would be found in the filter_file then generated password would be rejected and apg(1) will generate another one. min_substr_len specifies minimum substring length to check. This option is active only if -b option is defined.

Pseudo random number generator options

-s Ask user for random sequence for password generation

-c cl_seed

Use cl_seed as a random seed for password generation. I use it when i have to generate passwords in a shell script.

Examples:

-c /dev/urandom

-c /tmp/seed_file

Password output options

-d Do NOT use any delimiters between generated passwords. I use it when i have to generate passwords in a shell script.

-y Print generated passwords and crypted passwords (see crypt(3))

-q Quiet mode (do not print warnings)

-l Spell generated passwords. Useful when you want to read generated password by telephone.

WARNING: Think twice before read your password by phone.

-t Print pronunciation for generated pronounceable password. Ignored if -a 1 is set.

-h Print help information and exit

-v Print version information and exit

DEFAULT OPTIONS

apg -a 0 -M sncl -n 6 -x 10 -m 8 (new style)

If you want to generate really secure passwords, you should use option -s. To sim?

plify apg usage, you can write a small shell script. For example:

[begin]----> pwgen.sh

#!/bin/sh

/usr/local/bin/apg -m 8 -x 12 -s

[end]----> pwgen.sh

EXIT CODE

On successful completion of its task, apg will complete with exit code 0. An exit

code of -1 indicates an error occurred. Textual errors are written to the standard

error stream.

DIAGNOSTICS

If /dev/random is not available, apg will display a message about it.

FILES

None.

BUGS

None. If you've found one, please send bug description to the author.

SEE ALSO

apgbfm(1)

AUTHOR

Adel I. Mirzazhanov, <a-del@iname.com>

Project home page: http://www.adel.nursat.kz/apg/