## NAME

google-authenticator - initialize one-time passcodes for the current user

## SYNOPSIS

google-authenticator [*options*]

If no option is provided on the command line, google-authenticator(1) will ask interactively the user for the more important options.

## DESCRIPTION

The google-authenticator(1) command creates a new secret key in the current user's home directory. By default, this secret key and all settings will be stored in ~/.google_authenticator.

If the system supports the libqrencode library, a QRCode will be shown, that can be scanned using the Android Google Authenticator application. If the system does not have this library, google-authenticator(1) outputs an URL that can be followed using a web browser. Alternatively, the alphanumeric secret key is also outputted and thus can be manually entered into the Android Google Authenticator application.

In either case, after the key has been added, the verification value should be checked. To do that, the user must click-and-hold the added entry on its Android system until the context menu shows. Then, the user checks that the displayed key's verification value matches the one provided by google-authenticator(1). Please note that this feature might not be available in all builds of the Android application.

Each time the user logs into the system, he will now be prompted for the TOTP code (time based one-time-password) or HOTP (counter-based one-time-password), depending on options given to google-authenticator(1), after having entered its normal user id and its normal UNIX account password.

## OPTIONS

The main option consists of choosing the authentication token type: either time based or counter-based.

**-c, --counter-based**
Set up counter-based verification.

**-t, --time-based**
Set up time-based verification.

From this choice depends the available options.

**Counter-based specific options**

Those settings are only relevant for counter-based one-time-password (HOTP):

**-w, --window-size=***W*

Set window of concurrently valid codes.

By default, three tokens are valid at any one time. This accounts for generated-but-not-used tokens and failed login attempts. In order to decrease the likelihood of synchronization problems, this window can be increased from its default size of 3.

The window size must be between 1 and 21.

**-W, --minimal-window**

Disable window of concurrently valid codes.

**Time-based specific options**

Those settings are only relevant for time-based one-time-password (TOTP):

**-D, --allow-reuse, -d, --disallow-reuse**

(Dis)allow multiple uses of the same authentication token.

This restricts the user to one login about every 30 seconds, but it increases the chances to notice or even prevent man-in-the-middle attacks.

**-w, --window-size=***W*

Set window of concurrently valid codes.

By default, a new token is generated every 30 seconds by the mobile application. In order to compensate for possible time-skew between the client and the server, an extra token before and after the current time is allowed. This allows for a time skew of up to 30 seconds between authentication server and client.

For example, if problems with poor time synchronization are experienced, the window can be increased from its default size of 3 permitted codes (one previous code, the current code, the next code) to 17 permitted codes (the 8 previous codes, the current code, and the 8 next codes). This will permit for a time skew of up to 4 minutes between client and server.

The window size must be between 1 and 21.

**-W, --minimal-window**

Disable window of concurrently valid codes.

**-S, --step-size=***S*
Set interval between token refreshes to *S* seconds.

By default, time-based tokens are generated every 30 seconds. A non-standard value can be configured in case a different time-step value must be used.

The time interval must be between 1 and 60 seconds.

**General options**

**-s, --secret=***FILE*
Specify a non-standard file location for the secret key and settings.

**-f, --force**
Write secret key and settings without first confirming with user.

**-l, --label=***LABEL*
Override the default label in otpauth:// URL.

**-i, --issuer=***ISSUER*
Override the default issuer in otpauth:// URL.

**-Q, --qr-mode=***none*|*ansi*|*utf8*
QRCode output mode.

Suppress the QRCode output (*none*), or output QRCode using either ANSI colors (*ansi*), or Unicode block elements (*utf8*).

Unicode block elements makes the QRCode much smaller, which is often easier to scan. Unfortunately, many terminal emulators do not display these Unicode characters properly.

**-r, --rate-limit=***N*, **-R, --rate-time=***M*, **-u, --no-rate-limit**
Disable rate-limiting, or limit logins to N per every M seconds.

If the system isn't hardened against brute-force login attempts, rate-limiting can be enabled for the authentication module: no more than *N* login attempts every *M* seconds.

The rate limit must be between 1 and 10 attempts. The rate time must be between 15 and 600 seconds.

**-e, --emergency-codes=***N*
Generate *N* emergency codes.

A maximum of 10 emergency codes can be generated.

**-q, --quiet**
Quiet mode.

**-h, --help**
Print the help message.

**SEE ALSO**

The Google Authenticator source code and all documentation may be downloaded from <https://github.com/google/google-authenticator-libpam>.