NAME

pam_google_authenticator - PAM module for Google two-factor authentication

SYNOPSIS

pam_google_authenticator.so [secret=*file*] [authtok_prompt=*prompt*] [user=*username*] [no_strict_owner] [allowed_perm=*0nnn*] [debug] [try_first_pass|use_first_pass|forward_pass] [noskewadj] [no_increment_hotp] [nullok] [echo_verification_code]

DESCRIPTION

The **pam_google_authenticator** module is designed to protect user authentication with a second factor, either time-based (TOTP) or counter-based (HOTP). Prior logging in, the user will be asked for both its password and a one-time code. Such one-time codes can be generated with the Google Authenticator application, installed on the user's Android device. To respectively generate and verify those one-time codes, a secret key (randomly generated) must be shared between the device on which one-time codes are generated and the system on which this PAM module is enabled.

Depending on its configuration (see *options* section), this module requires that a secret file is manually set up for each account on the system. This secret file holds the secret key and user-specific options (see **google-authenticator**(1)). Unless the **nullok** option is used, authentication tries will be rejected if such secret file doesn't exist. Alternatively, a system administrator may create those secret files on behalf of the users and then communicates to them the secret keys.

OPTIONS

secret=*file*

Specify a non-standard file location for the secret file.

By default, the PAM module looks for the secret file in the .google_authenticator file within the home of the user logging in. This option overrides this location.

The provided location may include the following short-hands:

- **${USER}** that will be interpreted as the username.
- **${HOME}** and **~** that will be interpreted as the user's home directory.

authtok_prompt=*prompt*

Override default token prompt.

Note that if spaces are present in the provided prompt, the whole argument must be wrapped in square brackets.

user=*username*
Switch to a hard-coded user prior to doing any file operation.

no_strict_owner
Disable the check against the secret file's owner.

By default, the secret file must be owned by the user logging in. This option disables this check.

allowed_perm=*0nnn*
Override checked permissions of the secret file.

By default, the secret file must be readable only by its owner (ie. mode *0600*). This option allows a different mode to be specified for this file.

debug
Enable more verbose log messages in syslog.

try_first_pass|use_first_pass|forward_pass
Stacking options for this PAM module.

Because some PAM clients cannot prompt the user for more than just the password, the following stacking options may be used:

- **try_first_pass**: before prompting the user for the one-time code, this module first tries the previous stacked module's password in case that satisfies this module as well.

- **use_first_pass**: force this module to use a previous stacked modules password. With this option, this module will never prompt the user for the one-time code. Thus, if no valid one-time code is available, the user will be denied access.

- **forward_pass**: query the user for both the system password and the verification code in a single prompt. The system password is then forwarded to the next PAM module, which will have to be configured with either the **use_first_pass** option, or the **try_first_pass** option.

noskewadj
Don't adjust time skew automatically.

By default, the PAM module makes an attempt to compensate for time skew between the server and the device on which one-time passcodes are generated. This option disable this behavior.

Note that this option is only relevant for time-based (TOTP) mode.

no_increment_hotp
Don’t increment the counter for failed attempts.

In some circonstance, failed passwords still get an OTP prompt. This option disables counter incrementation is such situations.

Note that this option is only relevant for counter-based (HOTP) mode.

nullok
Allow users to log in without OTP, if they haven’t set up OTP yet.

During the initial roll-out process, all users may not have created a secret key yet. This option allows them to log in, even if the secret file doesn’t exist.

echo_verification_code
Echo the verification code when it is entered by the user.

MODULE TYPE PROVIDED

Only the **auth** module type is provided.

RETURN VALUES

PAM_SUCCESS
Either the provided one-time code is correct or is a valid emergency code.

PAM_IGNORE
This module is ignored.

PAM_AUTH_ERR
The provided one-time code isn’t correct and isn’t a valid emergency code, or an error was encountered.

EXAMPLES

The following lines may be used to enable this PAM module:

- auth required pam_google_authenticator.so no_increment_hotp # Make sure the counter (for HOTP mode) isn’t incremented for failed attempts.

- auth required pam_google_authenticator.so nullok # Allow users to log in if their secret files don’t

exist

- auth required pam_google_authenticator.so secret=/var/unencrypted-home/${USER}/.google_authenticator # Store secret files in a specific location

- auth required pam_google_authenticator.so [authtok_prompt=Your secret token:] # Use a specific prompt

- auth required pam_google_authenticator.so noskewadj # Don't compensate time skew automatically

SECURITY NOTES

For highest security, make sure that both password and one-time code are being requested even if password and/or one-time code are incorrect. This means that *at least* the first of pam_unix.so *(or whatever other module is used to verify passwords) and pam_google_authenticator.so* should be set as **required**, not **requisite**.

SEE ALSO

google-authenticator(1).

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