NAME

pam_extrausers − Module for libnss−extrausers authentication

SYNOPSIS

pam_extrausers.so [...]

DESCRIPTION

This is similar to the standard Unix authentication module pam_unix. But instead of using /etc/passwd and /etc/shadow, it uses /var/lib/extrausers/passwd and /var/lib/extrausers/shadow.

The account component performs the task of establishing the status of the user's account and password based on the following *shadow* elements: expire, last_change, max_change, min_change, warn_change. In the case of the latter, it may offer advice to the user on changing their password or, through the **PAM_AUTHTOKEN_REQD** return, delay giving service to the user until they have established a new password. The entries listed above are documented in the **shadow**(5) manual page. Should the user's record not contain one or more of these entries, the corresponding *shadow* check is not performed.

The authentication component performs the task of checking the users credentials (password). The default action of this module is to not permit the user access to a service if their official password is blank.

The password component of this module performs the task of updating the user's password. The default encryption hash is taken from the **ENCRYPT_METHOD** variable from */etc/login.defs*

The session component of this module logs when a user logins or leave the system.

Remaining arguments, supported by others functions of this module, are silently ignored. Other arguments are logged as errors through **syslog**(3).

OPTIONS

debug
Turns on debugging via **syslog**(3).

audit
A little more extreme than debug.

nullok
The default action of this module is to not permit the user access to a service if their official password is blank. The **nullok** argument overrides this default and allows any user with a blank password to access the service.

nullok_secure
The default action of this module is to not permit the user access to a service if their official password is blank. The **nullok_secure** argument overrides this default and allows any user with a blank password to access the service as long as the value of PAM_TTY is set to one of the values found in /etc/securetty.

try_first_pass
Before prompting the user for their password, the module first tries the previous stacked module's password in case that satisfies this module as well.

use_first_pass
The argument **use_first_pass** forces the module to use a previous stacked modules password and will never prompt the user − if no password is available or the password is not appropriate, the user will be denied access.

nodelay
This argument can be used to discourage the authentication component from requesting a delay should the authentication as a whole fail. The default action is for the module to request a delay−on−failure of the order of two second.

use_authtok
When password changing enforce the module to set the new password to the one provided by a previously stacked **password** module (this is used in the example of the stacking of the **pam_cracklib**

module documented below).

not_set_pass
This argument is used to inform the module that it is not to pay attention to/make available the old or new passwords from/to other (stacked) password modules.

nis
NIS RPC is used for setting new passwords.

remember=*n*
The last *n* passwords for each user are saved in /etc/security/opasswd in order to force password change history and keep the user from alternating between the same password too frequently. Instead of this option the **pam_pwhistory** module should be used.

shadow
Try to maintain a shadow based system.

md5
When a user changes their password next, encrypt it with the MD5 algorithm.

bigcrypt
When a user changes their password next, encrypt it with the DEC C2 algorithm.

sha256
When a user changes their password next, encrypt it with the SHA256 algorithm. If the SHA256 algorithm is not known to the **crypt**(3) function, fall back to MD5.

sha512
When a user changes their password next, encrypt it with the SHA512 algorithm. If the SHA512 algorithm is not known to the **crypt**(3) function, fall back to MD5.

blowfish
When a user changes their password next, encrypt it with the blowfish algorithm. If the blowfish algorithm is not known to the **crypt**(3) function, fall back to MD5.

rounds=*n*
Set the optional number of rounds of the SHA256, SHA512 and blowfish password hashing algorithms to *n*.

broken_shadow
Ignore errors reading shadow information for users in the account management module.

minlen=*n*
Set a minimum password length of *n* characters. The default value is 6. The maximum for DES crypt−based passwords is 8 characters.

obscure
Enable some extra checks on password strength. These checks are based on the "obscure" checks in the original shadow package. The behavior is similar to the pam_cracklib module, but for non−dictionary−based checks. The following checks are implemented:

Palindrome
Verifies that the new password is not a palindrome of (i.e., the reverse of) the previous one.

Case Change Only
Verifies that the new password isn't the same as the old one with a change of case.

Similar
Verifies that the new password isn't too much like the previous one.

Simple
Is the new password too simple? This is based on the length of the password and the number of different types of characters (alpha, numeric, etc.) used.

Rotated

Is the new password a rotated version of the old password? (E.g., "billy" and "illyb")

Invalid arguments are logged with **syslog**(3).

MODULE TYPES PROVIDED

All module types (**account**, **auth**, **password** and **session**) are provided.

RETURN VALUES

PAM_IGNORE
Ignore this module.

EXAMPLES

An example usage for /etc/pam.d/common−password would be:

```
password        [success=2 default=ignore]      pam_extrausers.so obscure sha512
password        [success=1 default=ignore]      pam_unix.so obscure sha512
# here's the fallback if no module succeeds
password        requisite                       pam_deny.so
# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success code
# since the modules above will each just jump around
password        required                        pam_permit.so
# and here are more per−package modules (the "Additional" block)
password        optional        pam_gnome_keyring.so
password        optional        pam_ecryptfs.so
```

SEE ALSO

login.defs(5), **pam.conf**(5), **pam.d**(5), **pam**(7)

AUTHOR

pam_extrausers was written by various people.