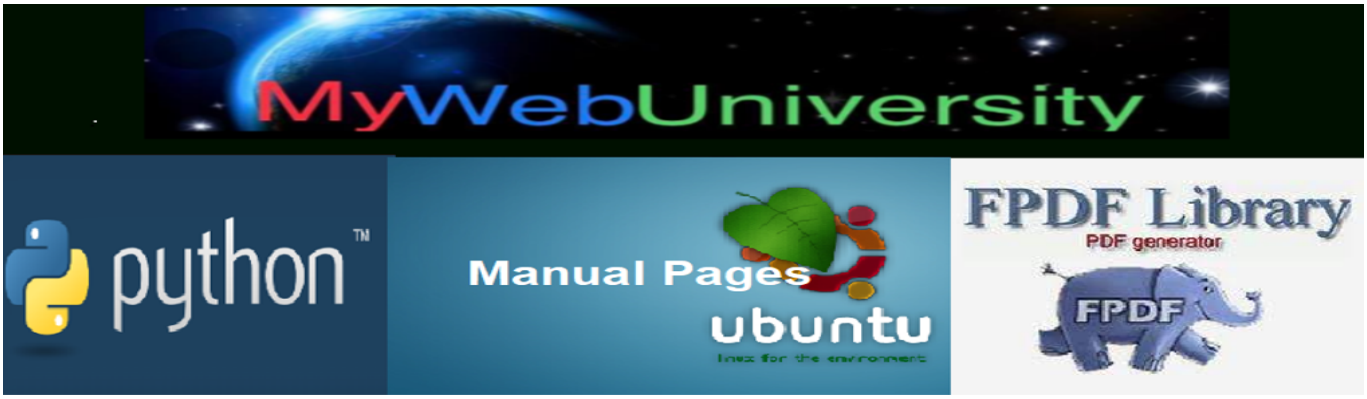




## *Rocky Enterprise Linux 9.2 Manual Pages on command 'persistent-keyring.7'*

***$ man persistent-keyring.7***

PERSISTENT-KEYRING(7) Linux Programmer's Manual PERSISTENT-KEYRING(7)

NAME

persistent-keyring - per-user persistent keyring

DESCRIPTION

The persistent keyring is a keyring used to anchor keys on behalf of a user. Each UID the kernel deals with has its own persistent keyring that is shared between all threads owned by that UID. The persistent keyring has a name (description) of the form _persis? tent.<UID> where <UID> is the user ID of the corresponding user.

The persistent keyring may not be accessed directly, even by processes with the appropri? ate UID. Instead, it must first be linked to one of a process's keyrings, before that keyring can access the persistent keyring by virtue of its possessor permits. This link? ing is done with the keyctl_get_persistent(3) function.

If a persistent keyring does not exist when it is accessed by the keyctl_get_persistent(3) operation, it will be automatically created.

Each time the keyctl_get_persistent(3) operation is performed, the persistent key's expi? ration timer is reset to the value in:

/proc/sys/kernel/keys/persistent_keyring_expiry

Should the timeout be reached, the persistent keyring will be removed and everything it pins can then be garbage collected. The key will then be re-created on a subsequent call to keyctl_get_persistent(3).

The persistent keyring is not directly searched by request_key(2); it is searched only if it is linked into one of the keyrings that is searched by request_key(2).

The persistent keyring is independent of clone(2), fork(2), vfork(2), execve(2), and

_exit(2). It persists until its expiration timer triggers, at which point it is garbage collected. This allows the persistent keyring to carry keys beyond the life of the ker? nel's record of the corresponding UID (the destruction of which results in the destruction of the user-keyring(7) and the user-session-keyring(7)). The persistent keyring can thus be used to hold authentication tokens for processes that run without user interaction, such as programs started by cron(8).

The persistent keyring is used to store UID-specific objects that themselves have limited lifetimes (e.g., kerberos tokens). If those tokens cease to be used (i.e., the persistent keyring is not accessed), then the timeout of the persistent keyring ensures that the cor? responding objects are automatically discarded.

### Special operations

The keyutils library provides the keyctl_get_persistent(3) function for manipulating per? sistent keyrings. (This function is an interface to the keyctl(2) KEYCTL_GET_PERSISTENT operation.) This operation allows the calling thread to get the persistent keyring corre? sponding to its own UID or, if the thread has the CAP_SETUID capability, the persistent keyring corresponding to some other UID in the same user namespace.

## NOTES

Each user namespace owns a keyring called .persistent_register that contains links to all of the persistent keys in that namespace. (The .persistent_register keyring can be seen when reading the contents of the /proc/keys file for the UID 0 in the namespace.) The keyctl_get_persistent(3) operation looks for a key with a name of the form _persis? tent.<UID> in that keyring, creates the key if it does not exist, and links it into the keyring.

## SEE ALSO

keyctl(1), keyctl(3), keyctl_get_persistent(3), keyrings(7), process-keyring(7), session-keyring(7), thread-keyring(7), user-keyring(7), user-session-keyring(7)

## COLOPHON

This page is part of release 5.10 of the Linux man-pages project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man-pages/.