36.5. `crypt` — Function to check Unix passwords¶

Platforms: Unix

This module implements an interface to the *crypt(3)* routine, which is a one-way hash function based upon a modified DES algorithm; see the Unix man page for further details. Possible uses include allowing Python scripts to accept typed passwords from the user, or attempting to crack Unix passwords with a dictionary.

Notice that the behavior of this module depends on the actual implementation of the *crypt(3)* routine in the running system. Therefore, any extensions available on the current implementation will also be available on this module.

`crypt.crypt`(*word*, *salt*)¶

word will usually be a user's password as typed at a prompt or in a graphical interface. *salt* is usually a random two-character string which will be used to perturb the DES algorithm in one of 4096 ways. The characters in *salt* must be in the set `[./a-zA-Z0-9]`. Returns the hashed password as a string, which will be composed of characters from the same alphabet as the salt (the first two characters represent the salt itself).

Since a few *crypt(3)* extensions allow different values, with different sizes in the *salt*, it is recommended to use the full crypted password as salt when checking for a password.

A simple example illustrating typical use:

```
import crypt, getpass, pwd

def login():
    username = raw_input('Python login:')
    cryptedpasswd = pwd.getpwnam(username)[1]
    if cryptedpasswd:
        if cryptedpasswd == 'x' or cryptedpasswd == '*':
            raise NotImplementedError(
                "Sorry, currently no support for shadow passwords")
        cleartext = getpass.getpass()
        return crypt.crypt(cleartext, cryptedpasswd) == cryptedpasswd
    else:
        return 1
```

Previous topic

[36.4. `grp` — The group database](#)

Next topic

[36.6. `dl` — Call C functions in shared objects](#)

This Page

- [Show Source](#)