Python - Administración de sistemas

Sumario

Ejecución de scripts por SSH

Veamos un ejemplo:

- Ayuda módulo sys
- Ayuda módulo os

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

import subprocess
import sys

HOST="usuario@192.168.1.136"

COMMAND="uname -a"

ssh = subprocess.Popen(["ssh", "%s" % HOST, COMMAND],
                       shell=False,
                       stdout=subprocess.PIPE,
                       stderr=subprocess.PIPE)
result = ssh.stdout.readlines()
if result == []:
    error = ssh.stderr.readlines()
    print >>sys.stderr, "ERROR: %s" % error
else:
    print result
```

Módulo os

Página oficial del módulo os.
El módulo os nos permite acceder a funcionalidades dependientes del Sistema Operativo. Sobre todo, aquellas que nos refieren información sobre el entorno del mismo y nos permiten manipular la estructura de directorios (para leer y escribir archivos.
Veamos un ejemplo:

```
#!/usr/bin/python3

import os

# Descubrir nuestro directorio de trabajo actual
dirActual = os.getcwd()
print("Directorio trabajo actual: {}".format(dirActual))

# Cambiar de directorio
os.chdir('/tmp')
dirActual = os.getcwd()
print("\nAhora estamos en el directorio: {}".format(dirActual))

# Crear un directorio si este no existe
##Pedimos el path del directorio a crear:
carpeta = str(input("\nIntroduce directorio a crear (ruta completa): "))
##Primero comprobamos si esta ya existe
if os.path.exists(carpeta):   # Si queremos ver si existe y es un directorio: os.path.isdir(carpeta)
    print("La carpeta {} ya existe".format(carpeta))
    exit(1)
```

```
else:
    os.system("mkdir {}".format(carpeta))
    print("Creamos la carpeta {}".format(carpeta))
    # Para ver un listado del directorio donde se creó ahora mismo esa carpeta
    cpath = carpeta.split("/")[0:-1] #Seleccionamos todas menos la creada
    dir_madre = os.sep.join(cpath) #Que, en el caso de linux es "/".join(cpath)
    print("Directorio madre : {}".format(dir_madre))
    listado = os.listdir(dir_madre)
    print("listado de los elementos de {}:".format(dir_madre))
    for elemento in listado:
        print(" - {}".format(str(elemento)))

exit(0)
```

Este módulo consigue que los scripts sean portables entre unos sistemas operativos y otros. Veamos un ejemplo:

```
$ python3
>>> import sys, os
>>> sys.platform
'linux'
>>> print(os.sep)
/
>>> mypath = os.sep.join ('primero', 'segundo', 'tercero')
>>> print(mypath)
primero/segundo/tercero

# En un sistema Windows el separador sería "\" y ya lo hace el script automáticamente
```

- Tabla con los métodos del módulo os.
- Ayuda interesante en Librosweb.

Módulo shutil

Página oficial módlo shutil

Módulo subprocess

Página oficial del módulo subprocess
Ejecutar comandos con el método **call**:

```
import subprocess

#Primer modo
directorio = ['mkdir', '/home/jefe/cosa1']
subprocess.call(directorio)

#Segundo modo
directorio = 'mkdir /home/jefe/cosa2'
subprocess.call(directorio, shell=True)
```

- Es mucho más cómoda la segunda opción (*shell=True*), pero más peligrosa pues los comandos se ejecutan directamente en la shell y el programa se puede hacer vulnerable a ataques de "inyección de shell":

```
>>> path = input('Inserta la ruta del directorio: ')
Inserta la ruta del directorio: ATENCIÓN; rm * -rf
>>> subprocess.call('ls ' + path, shell=True) # ¡No ejecutar!
```

Capturar la salida con **popen**:

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-

from subprocess import PIPE, Popen

comando = 'ls -lha /home/jefe/scripts'
proceso = Popen(comando, shell=True, stdout=PIPE, stderr=PIPE)
error_encontrado = proceso.stderr.read()
listado = proceso.stdout.read().decode("utf-8")
```

```
proceso.stdout.close()

if not error_encontrado:
    print(listado)
else:
    print("Se produjo el siguiente error:\n {}".format(error_encontrado))
```

Módulo shutil

Página oficial módulo shutil

Módulo platform

El módulo platform nos da información sobre el sistema en el que estamos trabajando. Lo mejor para descubrir su potencial es ver un ejemplo:

```
cat prueba.py
#!/usr/bin/python
# -*- coding: utf-8 -*-

import platform

profile = [
platform.architecture(),
platform.dist(),
platform.libc_ver(),
platform.machine(),
platform.node(),
platform.platform(),
platform.processor(),
platform.python_build(),
platform.python_compiler(),
platform.python_version(),
platform.system(),
platform.uname(),
platform.version(),
]

for info in profile:
print info
```

Siendo la salida:

```
('64bit', 'ELF')
('debian', '8.2', '')
('glibc', '2.4')
x86_64
debian
Linux-3.16.0-4-amd64-x86_64-with-debian-8.2

('default', 'Mar  1 2015 12:57:24')
GCC 4.9.2
2.7.9
Linux
('Linux', 'debian', '3.16.0-4-amd64', '#1 SMP Debian 3.16.7-ckt11-1+deb8u3 (2015-08-04)', 'x86_64', '')
#1 SMP Debian 3.16.7-ckt11-1+deb8u3 (2015-08-04)
```

--Volver