Mejorando el desempeño

Multiplexación de sockets

Temas

- ThreadingMixIn
- Servidor de chat usando select.select
- Multiplexación de un servidor web usando select.epoll

SocketServer

- Clase ThreadingMixIn, crea un nuevo hilo para cada cliente.

Referencia

- Python 3:

  https://docs.Python.prg/3/library/socketserver.html

aplic_Python_02_02.py

Ejemplo de la utilidad de ThreadingMixIn de la clase SocketServer en Python

Descripción

- Se plantea realizar un servidor que cree un hilo por cada cliente que solicita conexión.
- Esto no tiene que programarse, directamente la librería de ThreadingMixIn se encarga de realizarlo.

Funcionamiento

- El código crea un hilo servidor y lo lanza en el fondo.
- Después lanza tres clientes de prueba que le envían mensaje al servidor.
- En respuesta el servidor regresa el mensaje a los clientes.
- En el método handle( ) del servidor se puede ver que se imprime la información de cada uno de los hilos y se puede ver la diferencia para cada una de las conexiones.

Aplic_Python_02_03.py

Servidor de chat usando select.select

Descripción

- Cuando se pueden presentar cientos de conexiones en un servidor, el crear un hilo por cada una de las conexiones no siempre es viable.
- Debido a las limitaciones de memoria y poder de procesamiento, se necesita una técnica más eficiente para un gran número de cliente.
- Python ofrece el módulo select para solventar el problema.

Descripción

- Podemos escribir un chat eficiente usando el método select( ) que evita bloquear el envío y recepción de las llamadas.
- Usaremos los argumentos de -- nombre y – puerto, si el nombre es servidor fungirá como el servidor de chat.

Funcionamiento

- Al inicio de l módulo, definimos dos funciones: send( ) y receive().
- El servidor de chat utiliza estas utilerías para su funcionamiento.

aplic_python_02_04.py

Multiplexando un servidor web usando select.epoll

Descripción

- El módulo select en Python tiene algunas herramientas especificas.
- En una máquina Linux se tienen una utilería llamada epoll.
- Utiliza el núcleo del sistema y sondeará los eventos de la red e informará cada vez que algo ocurra.

Decripción

- Consiste de un servidor web que puede regresar una simple línea de texto ante cualquier conexión de navegador.
- La idea consiste en que, durante la inicialización del servidor web, hacemos una llamada a select.epoll( ) y registrar el descriptor de archivos de nuestro servidor para notificaciones de eventos.

Funcionamiento

- En el constructor del servidor web EpoolServer, se crea un servidor de socket y se conecta con un puerto determinado.
- El socket del servidor de hace no bloqueante (setblocking(0)).
- La opción TCP_NODELAY nos permite compartir datos sin retenerlos en el buffer (como ocurre con la conexión SSH).
- Después, la instancia select.epoll() se crea y el descriptor de archivos de sockets se pasa para ayudar en la monitorización.

Funcionamiento

- En el método run( ) del servidor web inicial la recepción de los eventos de sockets.
- Estos eventos se denotan con:
  - EPOLLIN: Este socket lee eventos.
  - EPOLLOUT: Este socket escribe eventos.

Referencias

- Chou, E. (2018). ***Mastering Python Networking.*** Ed. Packt. 2ª ed. Birmingham, U. K.
- Hillar G. (2018). ***RESTful Python Web Services.*** Ed. Packt. 2ª ed. Birmingham, U. K.
- Ratan A. y otros (2019). ***Python Network Programming.*** Ed. Packt . 1ª ed. Birmingham, U. K.