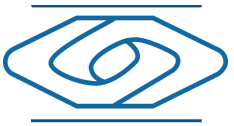
Instituto Politécnico Nacional
Escuela Superior de Cómputo
Desarrollo de Sistemas Distribuidos



Práctica #16

SDEP sincronización

(Diseño_SD_sincronización)

Pérez Herńandez Julio Alejandro
Hernández Estrada Luisa Anahi

4CM4

Ejercicio 1

Desarrollamos un programa utilizando las funciones gettimeofday, localtime y strftime para imprimir en un ciclo while infinito la hora exacta con precisión de microsegundos.

Ejecutamos en la terminal de dos computadoras distintas sincronizadas con un servidor NTP ¿Cuál es la diferencia de tiempo entre ambas computadoras?
hicimos varias pruebas y salieron resultados: 39970, 1059, 7072 microsegundos

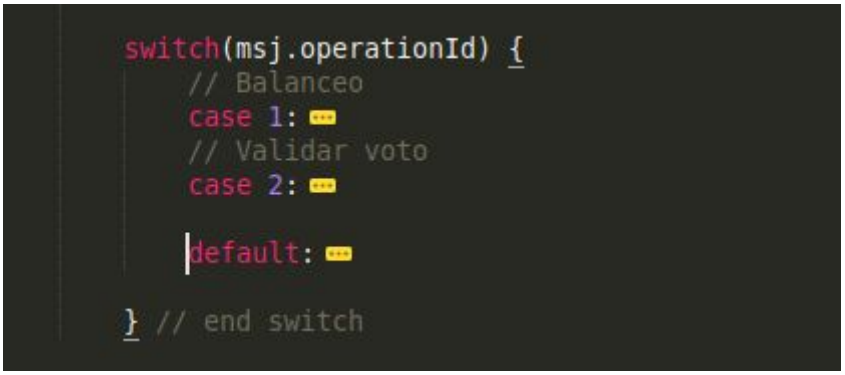
Ejecutamos en las terminales de la misma computadora. ¿Cuál es la diferencia de tiempo entre ambas terminales?
La diferencia fue de 14551 microsegundos

Ejercicio 2

Haciendo uso de los relojes lógicos vistos en la teoría, diseñe en equipo una solución al envío de timestamps únicos por parte de los servidores de votos ESCOM. Comente su solución con el profesor antes de implementar.

La solución que implementamos en esta parte fue que el servidor principal A (referenciando a la práctica anterior), tuviera tres funciones principales:

1. Balancear la carga entre servidores
2. Verificar votos
3. Responder al cliente



La primera función, **balancear la carga,** implica realizar el balanceo para el procesamiento de votos. Es decir, si se cumple la condición de balanceo (pares e impares + 0) se asigna una IP correspondiente al servidor A o B.

```
if((phone[9]-'0')%2!=0 || phone[9]=='0' /*|| phone[9]=='0'*/){
    // cout<<"Enviando al servidor 1"<<endl;
    const char *ipServer = "192.168.28.91";
} else {
    // cout<<"Enviando al servidor 2"<<endl;
    const char *ipServer = "192.168.28.98";
}
```

Una vez que balancea la carga, el servidor A ejecutar una función (por medio de hilos) que manda un mensaje tipo cliente al servidor con la IP **ipServer** pero cambiando el numero de operacion a 2, es decir a **la verificación de votos**.

```
void enviarServer(char *ipServer, struct mensaje msj, Respuesta res){
    struct timeval timeout;
    timeout.tv_sec = 2;
    timeout.tv_usec = 500000;
    char *ip = ipServer;
    int puerto = 2407;
    int operacion = 2;
    Solicitud cliente = Solicitud(timeout);
    char *respuesta = cliente.doOperation(ip, puerto, operacion, msj.arguments);
```

Por otra parte, la función **verificar votos**, realizar el procesamiento del voto como tal, busca que no esté repetido en la base de datos y envía respuesta con el mensaje de **confirmación o duplicado de voto.**

Finalmente, el servidor A obtiene la respuesta de verificación de voto y será el encargado de **responder al cliente** dicha respuesta, siendo este servidor quien les da un identificador **timestamp** y lo guarda en su propia base de datos.

- La implementación se encuentra en **servidor.cpp** en el método **enviarServer()**

Cabe mencionar que el cliente que lee el archivo de votos, envía todos estos al servidor A en el puerto X que balancea e identifica las respuestas con un timestamp. Este mismo servidor también ejecuta el mismo código en el puerto **2407** a fin de ser este el que valida los votos.