Memoria de P1

Introducción a ONC-RPC

Shunya Zhan

05 de Marzo del 2025

ÍNDICE

Introducción

El objetivo de este trabajo es implementar un sistema de comunicación cliente-servidor utilizando ONC-RPC (Open Network Computing Remote Procedure Call). ONC-RPC es un protocolo que permite a los programas ejecutar procedimientos en otras máquinas de la red como si fueran locales, facilitando la creación de aplicaciones distribuidas.

En este proyecto, se ha desarrollado un cliente y un servidor que se comunican mediante ONC-RPC. El servidor ofrece varios procedimientos remotos que el cliente puede invocar para obtener información sobre el servidor. Estos procedimientos incluyen la obtención del ID del proceso del servidor, el nombre del usuario que ejecuta el servidor, la dirección IP del servidor y el nombre del host del servidor.

El sistema está compuesto por los siguientes archivos principales:

- estu_cliente.c: Implementa el cliente que se conecta al servidor y llama a los procedimientos remotos.
- estu_servidor.c: Implementa el servidor que registra y ejecuta los procedimientos remotos.
- estu_cabecera.h: Define las constantes y los números de procedimiento utilizados por el cliente y el se rvidor.

El cliente y el servidor se comunican utilizando el protocolo ONC-RPC, que se encarga de la serialización y deserialización de los datos, así como de la transmisión de los mensajes a través de la red. El servidor registra los procedimientos remotos utilizando la función registerrpc, y el cliente los invoca utilizando la función callrpc

(**Nota**: En la carpeta también está proporcionada los cliente.c, servidor.c y cabecera.h que es la versión 1 cuyo funcionamiento está incorrecto/incompleto. Posteriormente se detalla la implementación del servidor y del cliente, así como la comunicación entre ambos, las instrucciones de compilación y ejecución, las pruebas realizadas y los resultados obtenidos.)

Descripción del Sistema

El sistema está basado en una arquitectura cliente-servidor, donde el cliente y el servidor se comunican utilizando el protocolo ONC-RPC. El cliente envía solicitudes de procedimientos remotos al servidor, y el servidor ejecuta estos procedimientos y devuelve los resultados al cliente.

El sistema está compuesto por los siguientes archivos principales:

- **estu_cliente.c**: Este archivo contiene la implementación del cliente. El cliente se conecta al servidor y llama a varios procedimientos remotos para obtener información sobre el servidor. Los procedimientos remotos incluyen la obtención del ID del proceso del servidor, el nombre del usuario que ejecuta el servidor, la dirección IP del servidor y el nombre del host del servidor.
- **estu_servidor.c**: Este archivo contiene la implementación del servidor. El servidor registra varios procedimientos remotos utilizando la función registerrpc. Estos procedimientos incluyen la obtención del ID del proceso del servidor, el nombre del usuario que ejecuta el servidor, la dirección IP del servidor y el nombre del host del servidor.
- **estu_cabecera.h**: Este archivo contiene las definiciones de constantes y números de procedimiento utilizados por el cliente y el servidor.

Flujo de trabajo:

Cliente:

1. El cliente se inicia y se le proporciona la dirección IP del servidor como argumento.
2. El cliente llama a varios procedimientos remotos en el servidor utilizando la función callrpc.
3. Los resultados de los procedimientos remotos se imprimen en la consola.

Servidor:

1. El servidor se inicia y registra varios procedimientos remotos utilizando la función registerrpc.
2. El servidor se pone a la escucha de solicitudes de procedimientos remotos utilizando la función svc_run.
3. Cuando el servidor recibe una solicitud de procedimiento remoto, ejecuta el procedimiento correspondiente y devuelve el resultado al cliente.

Implementación del Servidor

El servidor se encarga de registrar y ejecutar los procedimientos remotos que pueden ser invocados por el cliente. A continuación, se describe la implementación del servidor en el archivo estu_servidor.c

El servidor registra los procedimientos remotos utilizando la función registerrpc. Esta función toma como argumentos el número de programa, la versión del programa, el número de procedimiento, un puntero a la función que implementa el procedimiento, y dos funciones XDR para la serialización y deserialización de los argumentos y resultados.

- getprocessID: Obtiene el ID del proceso del servidor utilizando la función getpid.
- getuserNAME: Obtiene el nombre del usuario que ejecuta el servidor utilizando la función getlogin. La cadena devuelta por getlogin se duplica utilizando strdup para evitar la sobreescritura.
- getserverIP: Obtiene la dirección IP del servidor utilizando la función gethostbyname. La dirección IP se convierte a una cadena legible utilizando inet_ntoa.
- gethostnameSERVER: Obtiene el nombre del host del servidor utilizando la función gethostname. La cadena devuelta se duplica utilizando strdup.
- registerrpc: Registra un procedimiento remoto en el servidor.
- svc_run: Pone al servidor a la escucha de solicitudes de procedimientos remotos.
- getpid: Obtiene el ID del proceso actual.
- getlogin: Obtiene el nombre del usuario que ha iniciado sesión.
- gethostbyname: Obtiene información sobre un host a partir de su nombre.
- inet_ntoa: Convierte una dirección IP en formato binario a una cadena legible.
- gethostname: Obtiene el nombre del host.

Implementación del Cliente

El cliente se encarga de conectarse al servidor y llamar a los procedimientos remotos registrados en el servidor. A continuación, se describe la implementación del cliente en el archivo estu_cliente.c

El cliente se conecta al servidor utilizando la dirección IP proporcionada como argumento y llama a varios procedimientos remotos para obtener información sobre el servidor. Los resultados de estos procedimientos se imprimen en la consola.

- main: La función principal del cliente. Se encarga de verificar los argumentos de entrada, establecer la conexión con el servidor y llamar a los procedimientos remotos.
- call_remote_procedure: Esta función se encarga de realizar la llamada a un procedimiento remoto en el servidor. Toma como argumentos la dirección del servidor, el número del procedimiento, las funciones XDR para la serialización y deserialización de los argumentos y resultados, y el nombre del procedimiento.
- callrpc: Realiza una llamada a un procedimiento remoto en el servidor.
- clnt_perrno: Imprime un mensaje de error correspondiente al código de error devuelto por callrpc.
- xdr_void: Función XDR para la serialización y deserialización de argumentos vacíos.
- xdr_int: Función XDR para la serialización y deserialización de enteros.
- xdr_wrapstring: Función XDR para la serialización y deserialización de cadenas de caracteres.

Comunicación entre Cliente-Servidor

La comunicación entre el cliente y el servidor se realiza utilizando el protocolo ONC-RPC (Open Network Computing Remote Procedure Call). Este protocolo permite que el cliente invoque procedimientos remotos en el servidor como si fueran locales. A continuación, se describe cómo se lleva a cabo esta comunicación en el sistema.

Protocolo de comunicación

ONC-RPC utiliza un modelo de cliente-servidor en el que el cliente envía solicitudes de procedimientos remotos al servidor, y el servidor ejecuta estos procedimientos y devuelve los resultados al cliente. La comunicación se realiza a través de llamadas a procedimientos remotos (RPC).

Serialización y deserialización

ONC-RPC utiliza XDR (External Data Representation) para la serialización y deserialización de los datos. XDR es un estándar que define cómo representar datos en un formato independiente de la máquina, lo que permite la interoperabilidad entre sistemas heterogéneos.

Funciones clave

- callrpc: Esta función se utiliza en el cliente para realizar una llamada a un procedimiento remoto en el servidor. Toma como argumentos la dirección del servidor, el número de programa, la versión del programa, el número de procedimiento, las funciones XDR para la serialización y deserialización de los argumentos y resultados, y los propios argumentos y resultados.
- registerrpc: Esta función se utiliza en el servidor para registrar un procedimiento remoto. Toma como argumentos el número de programa, la versión del programa, el número de procedimiento, un puntero a la función que implementa el procedimiento, y las funciones XDR para la serialización y deserialización de los argumentos y resultados.
- svc_run: Esta función se utiliza en el servidor para ponerlo a la escucha de solicitudes de procedimientos remotos. El servidor entra en un bucle infinito esperando solicitudes de clientes.

Memoria

La primera versión del **servidor** (servidor.c) no funcionaba correctamente. Incluía un procedimiento adicional squaredroot que no era necesario para el objetivo del proyecto.

En la versión estu_servidor.c, se encontró un problema específico en la función char **gethostnameSERVER(). El problema estaba relacionado con la declaración del puntero result, que no estaba correctamente inicializado, lo que causaba errores de puntero y cadena de caracteres. Se corrigió la declaración del puntero result en la función gethostnameSERVER() (static char * result;)para asegurar que se inicializara correctamente y se duplicara la cadena devuelta por gethostname utilizando strdup.

La primera versión del **cliente** (cliente.c) no funcionaba correctamente debido a varios problemas:

- Estructura del código: La estructura del código era menos modular, con llamadas a procedimientos remotos directamente en el main, lo que dificultaba el mantenimiento y la extensión del código.
- Argumentos de entrada: Requería dos argumentos de entrada, lo que complicaba el uso del cliente.

En la versión estu_cliente.c, se realizaron varias mejoras para solucionar los problemas encontrados en la versión anterior:

- Modularidad: Al principio, se utilizaban varios if para controlar los diferentes procesos, y después creó la función call_remote_procedure para realizar las llamadas a procedimientos remotos, mejorando la modularidad y la mantenibilidad del código.
- Simplificación de argumentos: Se simplificó el uso del cliente requiriendo solo la dirección IP del servidor como argumento de entrada.

Compilación y ejecución

Compilación: Abre una terminal y navega al directorio donde se encuentran los archivos fuente. Luego, ejecuta el siguiente comando para compilar:

- gcc -o cliente -I /usr/include/tirpc estu_cliente.c -l tirpc
- gcc -o servidor -I /usr/include/tirpc estu_servidor.c -l tirpc -lm

Ejecución: En la terminal, ejecuta el servidor con el siguiente comando:

- ./cliente <IP_del_servidor>
- ./servidor

Prueba y resultado

Cliente:

```
rpcuserclient@937a38ebb92c:~/p1$ gcc -o cliente -I /usr/include/tirpc client
e.c -l tirpc
rpcuserclient@937a38ebb92c:~/p1$ ./cliente 172.17.0.3
Conectaremos con el servidor 172.17.0.3
[PROC-1] PID del proceso servidor:  217
[PROC-2] USERNAME del proceso servidor:  rpcuserserver
[PROC-3] IP del proceso servidor:  172.17.0.3
[PROC-4] Hostname del proceso servidor:  15e5bddb0ad8
```

Esta salida muestra los resultados de los procedimientos remotos llamados por el cliente, incluyendo el ID del proceso del servidor, el nombre del usuario que ejecuta el servidor, la dirección IP del servidor y el nombre del host del servidor.

Servidor:

```
rpcuserserver@15e5bddb0ad8:~/p1$ gcc -o servidor -I /usr/include/tirpc servi
dor.c -l tirpc -lm
rpcuserserver@15e5bddb0ad8:~/p1$ ./servidor
Hostname: 15e5bddb0ad8
Hostname: 15e5bddb0ad8
```

Conclusión

En este trabajo, se ha implementado un sistema de comunicación cliente-servidor utilizando ONC-RPC (Open Network Computing Remote Procedure Call). El sistema permite que el cliente invoque procedimientos remotos en el servidor para obtener información sobre el servidor, como el ID del proceso, el nombre del usuario, la dirección IP y el nombre del host.

Durante el desarrollo de este sistema, se han aprendido varias lecciones importantes:

- **Uso de ONC-RPC:** Se ha adquirido experiencia en el uso del protocolo ONC-RPC para la comunicación entre aplicaciones distribuidas. Esto incluye la serialización y deserialización de datos utilizando XDR, así como el registro y la invocación de procedimientos remotos.
- **Arquitectura cliente-servidor**: Se ha comprendido mejor la arquitectura cliente-servidor y cómo diseñar e implementar sistemas distribuidos que se comunican a través de la red.
- **Manejo de errores**: Se ha aprendido la importancia de manejar adecuadamente los errores en la comunicación de red y en las llamadas a procedimientos remotos para garantizar la robustez del sistema.

En conclusión, este trabajo ha proporcionado una valiosa experiencia en el desarrollo de sistemas distribuidos utilizando ONC-RPC.

Anexo

- Código fuente
- Campus Virtual
- Copilot