웹 기반 원격 소프트웨어 디버거의 설계

홍창호, 박홍성
강원대학교

Design and Implementation of a Remote Debugger based on Web

Chang-Ho Hong, Hong-Seong Park
Kangwon National University

Abstract - Linux 시스템에서 gdb와 gdbserver를 이용하여 원격지의 대상 프로세스에 대해 호스트 컴퓨터에서 디버깅이 가능하다. 이는 상대적으로 적은 리소스를 가진 원격지 컴퓨터나 임베디드 시스템에서 프로세스를 원격지의 시스템보다 풍부한 리소스를 지닌 호스트 컴퓨터를 통해서 디버깅을 수행함으로써 직접 디버깅을 수행할 때 원격지 시스템에 오는 부하를 줄일 수 있다는 장점이 있다[1]. 하지만 gdb와 gdbserver를 이용한 원격지 디버깅은 gdb가 동작하는 호스트 컴퓨터가 반드시 gdb가 운용 가능한 환경이어야 한다는 점과 디버깅을 수행하기 전 호스트 컴퓨터에 대한 환경 설정에 많은 시간과 노력이 필요하다는 점을 생각했을 때 효율적이지 못하다. 또한 gdb에서 기본적으로 제공하는 사용자 인터페이스가 CLI(Comman Line Interface) 라는 점은 익숙하지 않은 사용자에게 불편함을 안겨줄 수 있다.

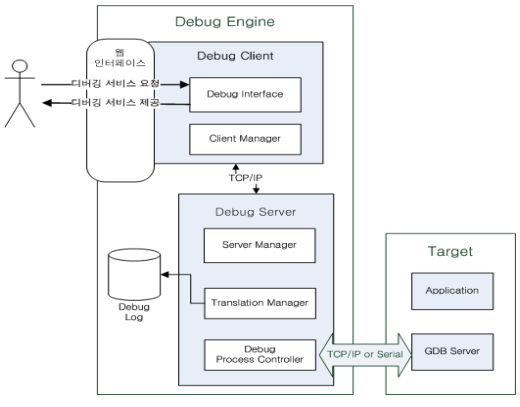
본 논문에서는 gdb와 gdbserver의 기본 동작을 웹 인터페이스와 연동함으로써 이러한 문제점에 대한 해결 방안을 제시하고자 한다.

1. 서 론

최근 Linux 기반의 시스템에서는 원격 디버깅 도구인 gdb와 gdbserver를 이용하여 원격지 시스템에서 동작하는 프로세스를 원격으로 디버깅할 수 있다. gdb는 호스트 시스템에서 실행시키고 원격지 시스템에서 gdbserver를 실행시킴으로써 상대적으로 호스트 시스템보다 부족한 자원을 갖는 타겟 시스템에서 동작하는 프로세스를 원격으로 디버깅 할 수 있도록 지원한다. gdb와 gdbserver를 사용한 원격 디버깅은 디버깅을 수행하고자 하는 사용자의 컴퓨터가 gdb가 운용 가능한 컴퓨터여야 한다는 점과 이를 구성하기 위해 사전 환경 설정에 많은 시간과 노력이 필요하게 된다. 또한, gdb의 기본 사용자 인터페이스가 CLI(Command Line Interface) 기반이므로 익숙하지 않은 사용자에게 원격지 디버깅을 수행할 때 불편함을 느끼게 할 수 있다. 본 논문에서는, gdb와 gdbserver의 동작을 웹과 연동함으로써 이를 해결하고자 하였다. 원격지 시스템에 접속하여 디버깅을 수행할 준비가 되어 있는 시스템을 항상 준비하여 두고, 사용자는 사용자의 컴퓨터에서 웹 브라우저로 접속함으로써 원격지의 프로세스에 대해 디버깅을 수행할 수 있게 된다. 이러한 웹 기반의 원격 디버깅 시스템을 구현하기 위한 시스템의 구조를 제시하도록 하겠다.

2. 웹 기반 원격 디버깅을 위한 시스템 구조

웹 기반 원격 디버깅을 위한 시스템의 기본 구조는 그림 1과 같다.



<그림 1> 웹 기반 원격 디버깅을 위한 시스템 전체 구조

사용자는 웹 인터페이스를 통하여 웹 기반 원격 디버깅 시스템을 통해 원격지 시스템의 프로세스에게 디버깅 연결을 요청하고 응답을 확인함으로써 본 시스템에 접근할 수 있다. 정상적으로 디버그 엔진 및 원격지 시스템에 연결이 이루어지면 웹 브라우저를 통해 브레이크 포인트 설정 및 해제, 프로세스 실행 등의 서비스 요청을 하고 디버깅 결과를 다시 웹 브라우저를 통해 확인할 수 있다.

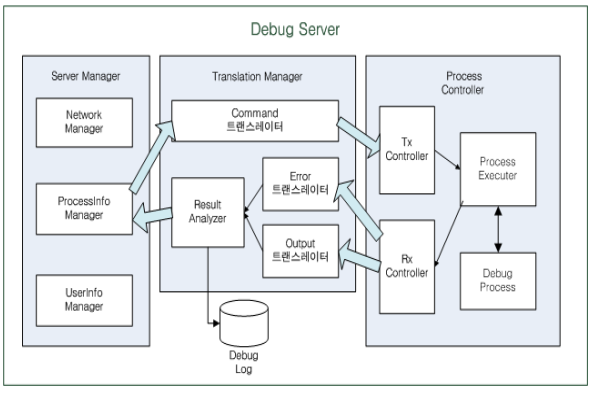
웹 기반 원격 디버깅을 위한 시스템의 기본 구조는 크게 2가지 구조로 구분되는데 웹 인터페이스와 디버그 클라이언트, 디버그 서버를 포함하는 디버그 엔진과 실제 디버깅 대상이 되는 응용 프로세스와 gdbserver가 동작하는 원격지 시스템이 있다.

2.1 디버그 엔진

디버그 엔진은 사용자로부터의 서비스 요청 및 응답이 이루어지는 웹 인터페이스와 디버그 클라이언트, 디버그 서버로 구성된다. 디버그 엔진은 서버-클라이언트 구조로 이루어지는데 이는 웹 인터페이스로부터 다중의 사용자가 접근할 경우 사용자 각각의 프로파일을 관리하기 위함이다[2].

디버그 클라이언트는 웹 인터페이스를 통해 입력된 사용자 요청을 실제 디버그 서버와의 소켓 통신을 위한 프로토콜로 전환시키기 위한 모듈인 디버그 인터페이스와 사용자의 프로파일 등을 관리하고 디버그 서버와의 연결 상태를 통제하고 관리하는 클라이언트 매니져 모듈로 이루어진다.

디버그 서버는 클라이언트 매니져와 동일한 역할을 수행하는 서버 매니져 모듈과 디버그 클라이언트로부터 전송된 요청 메시지를 실제 gdb 프롬프트에서 사용가능한 gdb 명령어로 전환시켜 주고, gdb를 통해 디버깅 된 결과를 사용자에게 보여주기 위한 형태로 전환시켜주는 트랜잭션 모듈 및 실제로 gdb 프로세스를 관리하고 gdb 실행 명령 및 디버깅 명령어를 입력하고, 그 결과를 전달받는 디버그 프로세스 컨트롤러 모듈로 구성된다. 디버그 서버의 세부 구조 및 데이터 흐름은 그림 2와 같다.



<그림 2> 디버그 서버의 구조 및 흐름

2.2 원격지 시스템

원격지 시스템은 실제 디버깅 대상이 되는 응용 프로세스가 용용되는 시스템으로 리눅스가 운영체제인 PC(Personal Computer)나 임베디드 시스템이 될 수 있다. 원격지의 gdbserver와 함께 동작시킴으로써 호스트 컴퓨터(디버그 엔진이 동작하는 컴퓨터)와 TCP/IP나 직렬 통신으로 연결되어 호스트 컴퓨터로부터 명령어를 전송 받아 원격지 시스템의 대상 프로세스를 디버깅 할 수 있다.

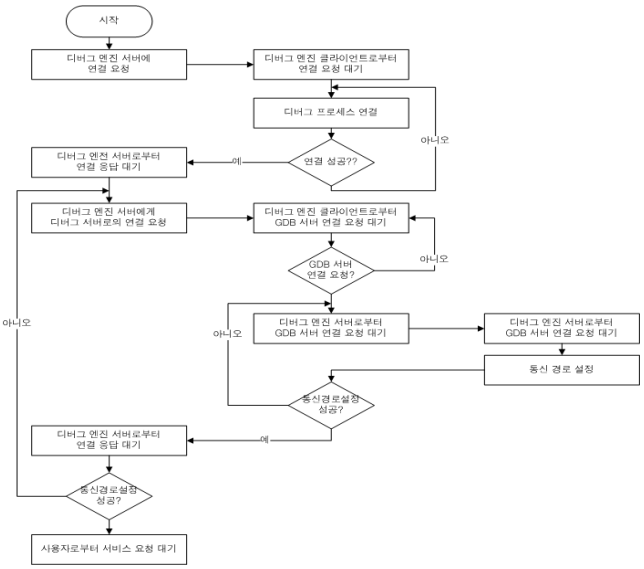
3. 웹 기반 원격 디버깅을 위한 시스템의 동작

본 시스템의 동작은 웹 인터페이스를 통한 디버그 엔진 및 타겟 시스템으로의 연결과 정상적인 연결 수 디버깅 서비스 및 응답으로 구분할 수 있다.

3.1 디버그 엔진 및 원격지 시스템과의 연결

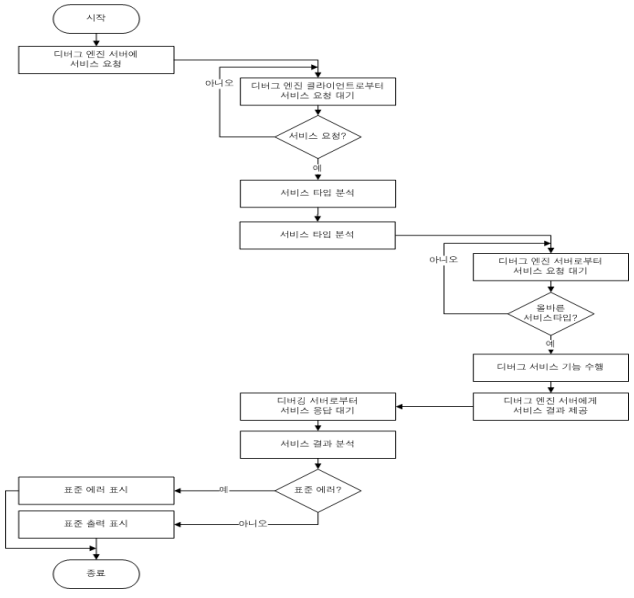
디버그 엔진 및 원격지 시스템의 연결은 그림 3과 같은 순서에 따라 이루어진다.

디버그 클라이언트에 연결 요청이 들어오면 디버그 클라이언트는 디버그 서버에게 연결을 요청하고 대기한다. 연결 요청을 받은 디버그 서버는 디버그 프로세스 컨트롤러 모듈을 통해 gdb에 대한 프로세스를 얻어 원격지 시스템의 gdbserver와 연결을 시도한다. 통신 경로가 설정되면 gdb와 gdbserver 간의 연결이 확립된 것이고, 그렇지 않으면 실패한 것이다. 통신 경로의 설정이 성공하면 디버그 클라이언트는 사용자로부터 서비스 요청을 대기 하고 그렇지 않으면 디버그 연결 요청 상태에서 대기하게 된다.



<그림 3> 원격지 시스템과의 연결 확립 과정

3.3 서비스 요청 및 응답



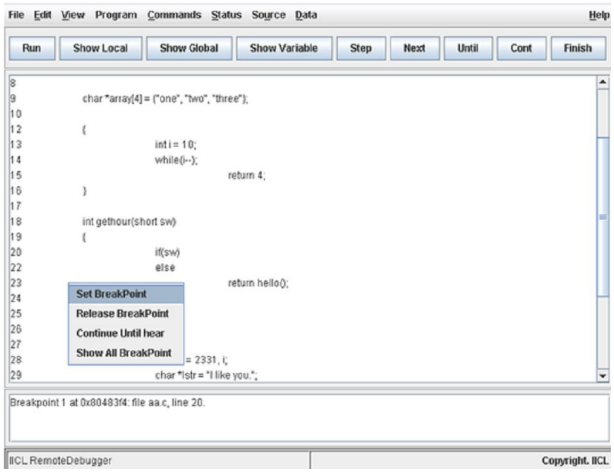
<그림 4> 서비스 요청 및 응답 순서도

그림 3에서 디버그 클라이언트가 서비스 요청 상태에서 대기하다가 사용자로부터 서비스 요청이 들어오면 디버그 클라이언트는 디버그 서버에게 이를 전달한다. 디버그 서버는 먼저 연결 요청인지 서비스 요청인지 확인 후 서비스 요청일 경우 서비스 타입을 분석하고 명령어 분석 후 디버그 프로세스 컨트롤러를 통해 gdb에게 전달한다. gdb가 명령어를 전달 받으면 gdbserver에게 전송하고 원격지 시스템의 대상 프로세스를 디버깅하여, 디버깅 결과를 gdb에게 응답하게 되고 gdb는 웹 브라우저를 통해 사용자에게 대상 프로세스의 디버깅 결과를 출력한다.

원격지 시스템에 연결 후 디버깅 서비스 요청 및 응답에 대한 순서도는 그림 4와 같다.

4. 구 현

본 시스템의 구현을 위해서 3대의 PC가 사용되었다. 한 대는 사용자 환경의 PC, 한 대는 디버그 엔진이 동작하는 PC, 나머지 한 대는 원격지 시스템으로 동작하는 PC이다. 디버그 엔진이 동작하는 PC와 원격지 시스템으로 동작하는 PC는 Linux 커널 2.6의 운영체제를 사용하였고, 사용자 환경의 PC는 WindowXP 서비스 팩 3 운영체제를 사용하였으며, 웹 브라우져로 Explorer 7을 사용하였다. 그림 5는 구현의 결과를 나타내고 있으며, 원격지의 해당 프로세스에 대해 브레이크 포인트 설정 및 해제, 프로세스 실행 및 중지 등의 디버깅을 수행하기 위한 기능들을 자바 애플릿을 이용하여 구현한 사용자 인터페이스를 통해 이루어진다.



<그림 5> 구현 화면

5. 결 론

본 논문에서는 웹 기반의 원격 디버깅을 위한 시스템의 구조를 제안하였다. 이를 이용하여 사용자는 웹 브라우저를 통해 원격지 컴퓨터의 프로세스에 대한 디버깅이 가능하도록 하여 효율적인 디버깅이 가능할 것으로 생각된다. 사용자는 자신의 컴퓨터 환경에 상관없이 웹 브라우져를 통해 미리 준비된 컴퓨터에 접속하여 원격지의 프로세스에 대해 디버깅을 수행할 수 있다. 또한, gdb의 기본 사용자 인터페이스가 CLI(Command Line Interface)이기 때문에 이에 익숙하지 않은 사용자에게 좀 더 편리한 디버깅 환경을 제공할 수 있을 것으로 기대된다.

[참 고 문 헌]

[1] 엄영익 외 2명, “Linux 시스템 기반 다중 프로세스 동시 디버깅을 지원하는 원격 디버거의 설계 및 구현”, KIPS, 2003

[2] 서영애, “이기종 환경을 지원하는 분산디버거의 설계 및 구현”, 대한전자공학회, 제 21 권, 523-526, 1998