공학 연구논문집 제28권 1호 pp. 287~298. 울산대학교.
Journal of Engineering Research Vol. 28, No. 1. pp. 287~298, 1997
University of Ulsan.

방화벽을 통한 X 응용프로그램의 지원에 관한 연구

이동헌 · 홍창열 · 김영곤 · 이명준
전자계산학과

〈요 약〉

인터넷은 매우 편리한 전산망이지만, 항상 안전한 상태는 아니다. 많은 조직들이 인터넷과 같은 공개 전산망상에서 다른 조직과 접속할 때 관리상의 보안 정책을 필요로 한다. 방화벽은 신뢰할 수 없는 전산망으로부터 하나의 전산망을 보호하는 방법중 하나이다. 방화벽은 내부 전산망과 외부 전산망사이의 트래픽(traffic)을 막거나 선택적으로 허용하는 기계장치를 요구한다. 방화벽은 또한 Ftp, Telnet과 같은 몇몇 제한된 인터넷 접근을 허용한다. 일반적 방화벽 시스템을 통해서는 외부 전산망상의 X 응용프로그램이 내부 전산망의 X 디스플레이에 접근하지 못하도록 되어있어 외부 전산망의 X 응용프로그램의 실행이 지원되지 않는다. 본 논문에서는 이러한 제약 조건을 극복하고 방화벽을 통해 X 응용프로그램 실행을 효과적으로 지원하는 기법을 제시한다.

Supporting X Applications through the Firewall

Dong Hun Lee · Chang Youl Hong · Young Gon Kim · Myoung Joon Lee
Dept. of Computer Science

<Abstract>

The Internet is a very convenient network, but not always a safe place. Many organizations often require administrative security policies when they want to connect with other organizations through a public network such as the Internet. A firewall is one way of protecting an internal network from the untrusted external network with a mechanism for selectively permitting or blocking traffic between the internal network and the external network. The firewall also provides some limited Internet access such

as Ftp, Telnet, etc. Through the usual firewall system, X applications running on the external network may not access an X display on the internal network. Thus, X applications running on the external network are not supported through the usual firewall. In this paper, we propose a method of supporting X application programs effectively through the firewall, overcoming this restriction.

1. 서 론

현대 사회에서 인터넷은 거의 모든 나라가 이용하고 있는 전산망이다. 인터넷을 사용하는 인구도 해마다 증가하고 있으며, 사용계층도 다양해지고 있다. 또한 인터넷은 세계를 하나로 묶어주는 역할도 하고 있다. 하지만 바이러스(virus), 웜(worm), 트로이목마(trojan horse), 몰래삽입하기(trap door), 눈속임(spoof)등과 같은 방법으로 침입하는 해커(hacker), 크래커(cracker), 인트루더(intruder)로부터 우리의 내부 전산망이 안전하다고는 할 수 없다[1]. 이 때문에 우리의 내부 전산망을 외부 전산망으로부터 보호할 수 있는 보호 정책이 필요하다. 이러한 필요성에 의해 구성되어진 것이 바로 **방화벽**(firewall) 시스템이다[1].

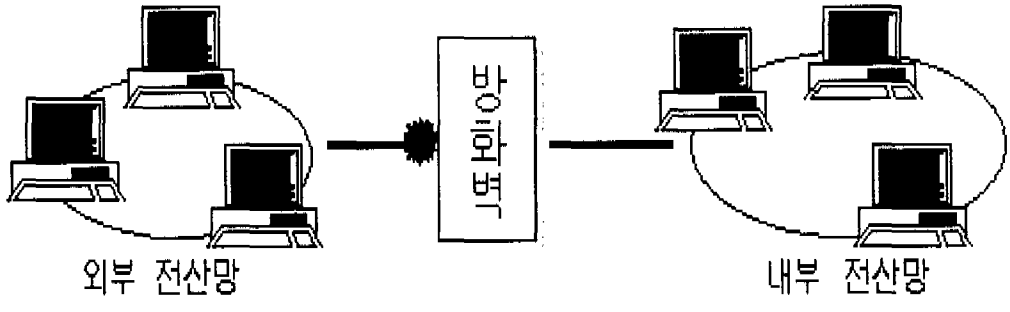


그림 1. 방화벽

그림 1과 같이 방화벽은 내부 전산망과 외부 전산망 사이에 설치되어 외부 전산망으로부터 내부 전산망을 보호하는 역할을 한다. 일반적으로 방화벽은 **라우터**(router)와 **베스천 호스트**(bastion host)로 구성된다. 라우터는 내부와 외부 전산망을 분리하는데 이용되고, 베스천호스트는 외부 전산망과 내부 전산망 모두에 접속 되어 있는 호스트이다.

방화벽에서 이용되어지는 스크리닝 라우터(screening router)는 내부와 외부 시스템이 직접 접근을 할수 있도록 하는 장치이다[2]. 스크리닝 라우터는 방화벽 시스템을 보호하고, 방화벽 시스템과 내부 시스템 사이의 상호 동작을 제한(내부 시스템이 방화벽을 이용하는 수를 제한)하며, 베스천 호스트는 내부 전산망과 외부 전산망 모두에 연결되어 있으며 내부 전산망에서 외부 전산망으로 여러가지 다양한 서비스를 제공할 수 있도록 한다[3]. 또한 방화벽 시스템에는 누가 접속되어 있는가, 어떤 종류의 자료흐름(방화벽 안으로 혹은

밖으로)이 발생하는가, 또는 어떤 응용프로그램이 이용되고 있는가에 대한 관리 정책도 포함하고 있다. 이때 스크리닝 라우터가 상당한 보호기능을 제공한다. 그러나 스크리닝 라우터만으로는 많은 응용프로그램에 대한 완벽한 사용을 지원할 수 없으므로 베스천호스트에 응용 프로그램을 서비스해 줄수 있는 소프트웨어를 설치하여 이용하고 있다. 하지만 X 윈도우 시스템을 이용하는 응용프로그램은 일반적 방화벽 구조에서는 지원해 주지 않는다. X 윈도우 시스템에서는 사용자가 환경변수 DISPLAY를 변경함으로써 X 디스플레이 서버를 변경할 수 있다. 하지만 방화벽의 외부 전산망에서는 내부 전산망에 대한 호스트 정보를 알수 없다. 그러므로 외부 전산망의 **원격 호스트**(remote host)에서 현재 자신이 사용중인 디스플레이로 X 디스플레이 서버를 변경할 수가 없다. 따라서 본 논문에서는 방화벽 시스템에서 X 응용프로그램 실행을 지원할 수 있도록 *xrelay* 서버와 클라이언트를 구현하였다.

xrelay는 방화벽 시스템에 설치할 서버 프로세스인 *xrelayd*와 외부 전산망의 원격 호스트에서 서비스를 요청할 때 사용할 클라이언트 프로세스인 xrelay의 두부분으로 구성된다. 서버 프로세스인 xrelayd는 *daemon*으로서 클라이언트 프로세스인 xrelay로부터의 요청이 있을때 자식프로세스를 생성하여 원격 호스트와 디스플레이 사이에서 서비스를 해주게 된다. 클라이언트 프로세스인 xrelay는 원격 호스트에서 서비스를 받을 방화벽의 인터넷 이름과 방화벽 내부 전산망의 디스플레이를 명시하여 서비스를 요청한 후 방화벽의 서버로부터의 응답이 오면 자신은 원격 호스트의 **흠쉘**로 바뀌어 동작하게된다.

다음장은 방화벽 시스템의 구성형태에 따른 종류와 방화벽의 역할에 대하여 기술을 한다. 그리고 3장에서는 xrelay의 사용법에 관하여 설명을 하며 4장에서는 xrelay의 설계 및 구현방법에 대하여 기술을 한다. 그리고 5장에서는 관련 연구를, 6장에서는 결론및 추후 연구방향에 관해 차례로 기술한다.

2. 방화벽 시스템

2.1 시스템 보안책

인터넷을 통해 침입하는 침입자로부터 시스템을 보안하기 위한 방법에는 여러가지가 다. 그중 계정 보안, 패스워드 보안, 파일시스템 보안등이 지금까지 사용되어온 대표적인 방법이다.

계정 보안 방법은 계정을 생성하고, 유지 현황를 관리하고, 이용자의 정상적인 사용을 확인하며, 이용자가 사용한 명령어와 프로세스를 감시한다. 또한 이용 현황 파악을 위한 기록 파일을 유지하여 불법 침입을 감시한다. 패스워드 보안 방법은 패스워드 형식 선택과 패스워드 파일의 형태 선택에서부터 패스워드를 어떻게 관리할 것이며, 어떤 패스워드 프로그램을 사용할 것인가에 대한 보안을 한다. 그리고 파일시스템 보안 방법에는 파일 억세스 허용 범위를 제한하고, 중요한 파일 시스템을 백업하고, 파일 내용을 암호화하는 것등이 있다. 하지만, 이러한 방법만으로는 내부 전산망 시스템을 안전하게 보안할 수 없으므로 더욱 효과적인 보안책으로 방화벽 시스템이 개발되어 이용되고 있다.

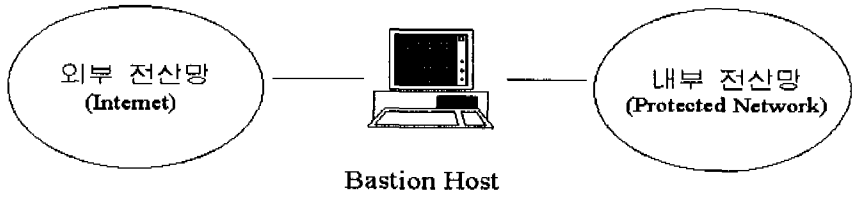
2.2 방화벽의 구성 형태

대부분의 방화벽은 스크리닝 라우터와 베스천 호스트로 구성된다. 스크리닝 라우터는 네트워크 레벨에서 선택적으로 트래픽(traffic)을 제어하는 방화벽의 보안 부분에 이용되어진다. 베스천 호스트는 방화벽 소프트웨어가 동작하는 호스트로서 게이트웨이 호스트(Gateway Host)라고도 불리운다. 베스천 호스트에는 내부 전산망에서 외부 전산망으로 이용할 수 있는 여러 가지 서비스를 제공한다. 스크리닝 라우터와 베스천 호스트의 구성에 따라 방화벽은 호스트 기반 방화벽(Host-based Firewall), 라우터 기반 방화벽(Router-based Firewall), 이중 네트워크 게이트웨이(Dual-Homed Gateway), 스크린 호스트 게이트웨이(Screened Host Gateway), 스크린 서브넷 게이트웨이(Screened Subnet Gateway)등으로 나눌수 있다.

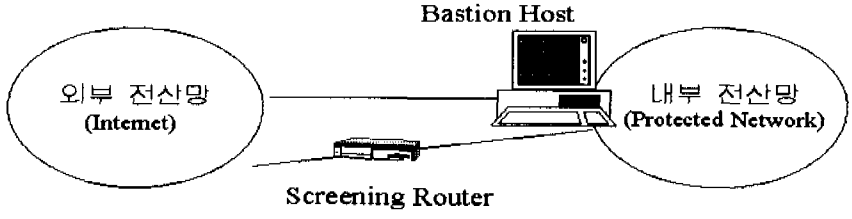
이중 네트워크 게이트웨이(Dual-Homed Gateway)는 그림 2.(a)와 같이 외부 전산망과 보안이 필요한 내부 전산망 양쪽에 연결된다. 외부 전산망과 내부 전산망이 연결되기 위해서는 항상 이 호스트를 거치도록 설계된다. 예를 들어, 인터넷상의 어떤 호스트에서 방화벽 내부의 호스트에 로그인하려할때는 항상 이 방화벽의 게이트웨이 호스트에 로그인 해야하고, 반대로 내부 전산망의 호스트도 인터넷상의 외부 호스트로 로그인하려 할때는 방화벽의 호스트에 먼저 로그인해야한다.

스크린 호스트 게이트웨이(Screened Host Gateway)는 방화벽 시스템을 내부 전산망에 두는 형태이다. 스크리닝 라우터는 외부 전산망에서 내부 전산망으로 들어가는 모든 트래픽(traffic)을 방화벽이 설치되어 있는 호스트에게만 전달되도록 하고, 내부에서 외부로 가는 모든 트래픽은 이 호스트에서 나온것만 허용한다. 그 형태는 그림 2.(b)와 같다.

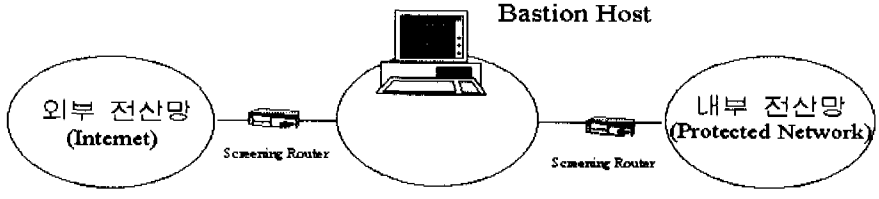
스크린 서브넷 게이트웨이(Screened Subnet Gateway)는 스크린 호스트 게이트웨이와 비슷하지만, 방화벽 시스템으로 하나의 전산망을 형성하여 내부 전산망와 외부 전산망 사이에 설치된다는 점이 스크린 호스트 게이트웨이와 다르다. 그림 2.(c)가 스크린 서브넷 게이트웨이 형태이다.



(a)이중 네트워크 게이트웨이



(b)스크린 호스트 게이트웨이



(c)스크린 서브넷 게이트웨이

그림 2. 방화벽의 구성형태

3. xrelay 시스템 사용법

그림 3과 같이 먼저 원격 호스트와 X 디스플레이 서버 사이의 중개자 역할을 할 서버 프로그램 xrelayd를 방화벽에 설치한다. xrelayd는 원격 호스트의 클라이언트 프로그램으로부터의 접속을 기다리며 접속이 이루어지면 자식 프로세스를 생성하여 원격 호스트와 **X 디스플레이 서버**(X display server) 사이에서 X 응용프로그램에 대한서비스를 해주게 된다.자세한내용은4.2에서언급된다.

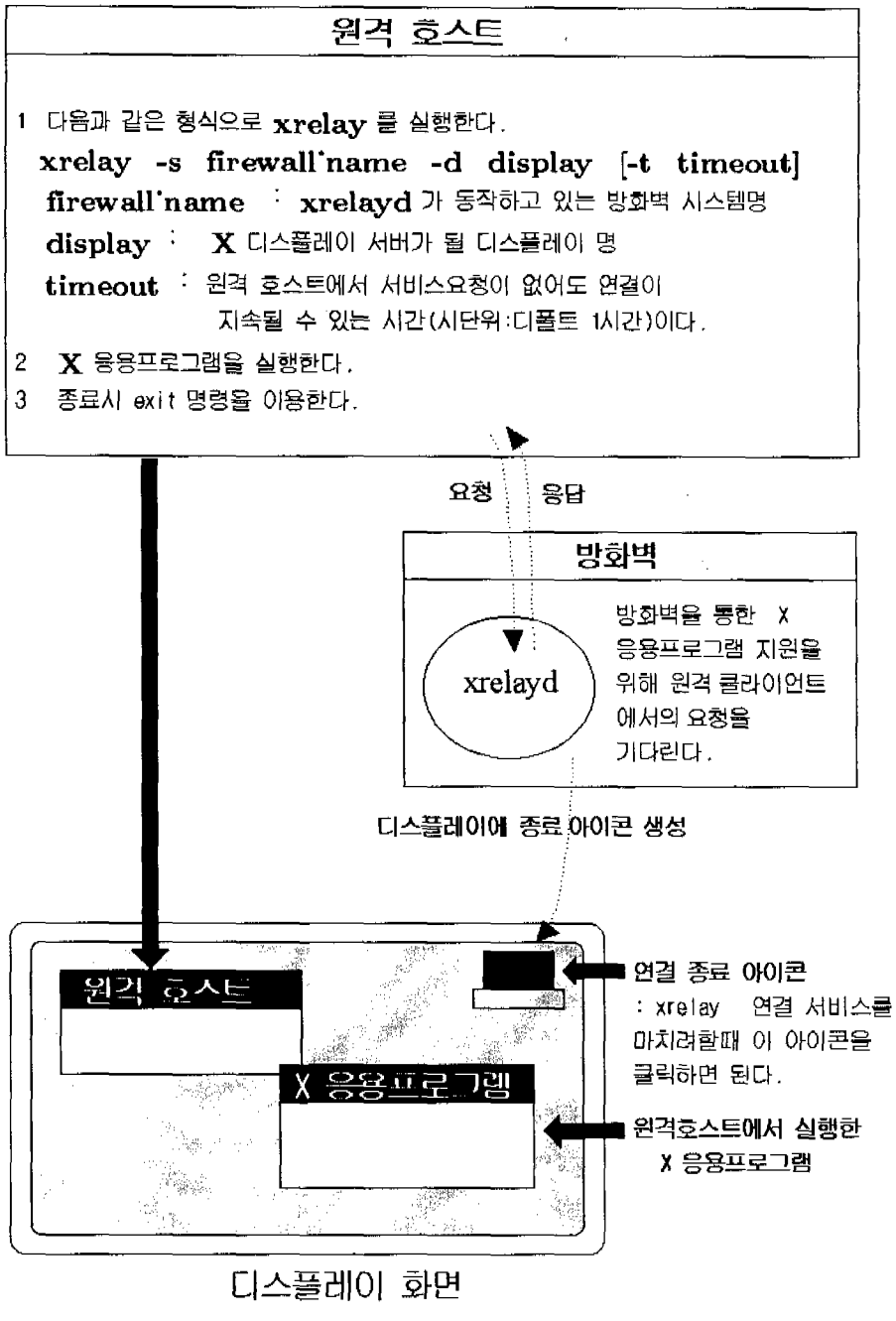

원격 호스트
1 다음과 같은 형식으로 xrelay 를 실행한다.
xrelay -s firewall'name -d display [-t timeout]
firewall'name : xrelayd 가 동작하고 있는 방화벽 시스템명
display : X 디스플레이 서버가 될 디스플레이 명
timeout : 원격 호스트에서 서비스요청이 없어도 연결이 지속될 수 있는 시간(시단위:디폴트 1시간)이다.
2 X 응용프로그램을 실행한다.
3 종료시 exit 명령을 이용한다.
요청
응답
방화벽
xrelayd
방화벽을 통한 X 응용프로그램 지원을 위해 원격 클라이언트에서의 요청을 기다린다.
디스플레이에 종료 아이콘 생성
원격 호스트
X 응용프로그램
연결 종료 아이콘
: xrelay 연결 서비스를 마치려할때 이 아이콘을 클릭하면 된다.
원격호스트에서 실행한 X 응용프로그램
디스플레이 화면

그림 3. 프로그램실행

원격 호스트에서 서버 프로세스로부터 서비스를 받기위해서는 먼저 그림 3의 원격 호스트에서 xrelay를 수행한다. firewall_name은 서버 프로세스가 위치해 있는 방화벽명이며, display는 X 디스플레이 서버의 디스플레이명이다. xrelay를 실행한 후 xrelay가 서버와 접속 되었다면 에러 메세지 없이 쉘 프롬프트가 뜨게된다. 이후의 X 윈도우 시스템에 관련된 작업은 모두 서버 프로세스 xrelayd가 서비스를 해주게 된다. 작업을 종료하고 싶으면 exit명령을 이용한다.

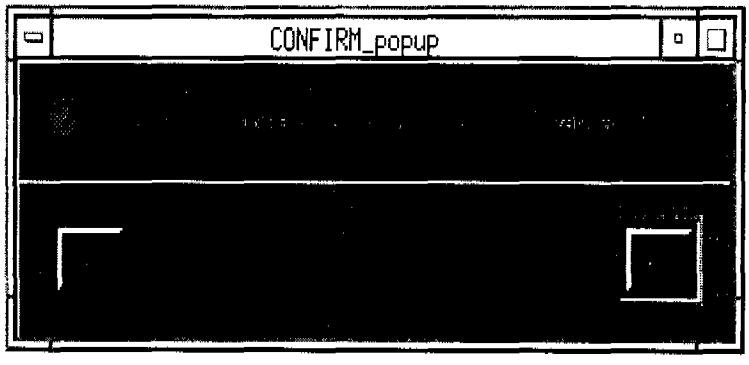


그림 4. 사용인증허가 윈도우

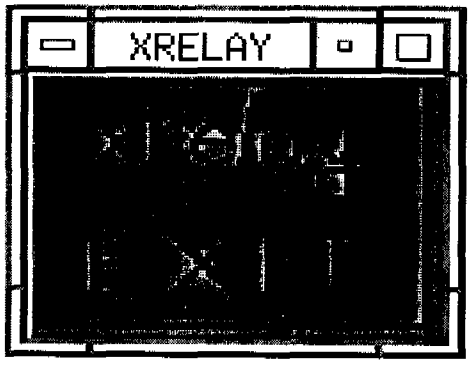


그림 5. 연결종료 아이콘

원격 호스트에서 xrelay를 실행한 후, X 응용프로그램을 실행시키면 X 디스플레이 서버 화면에 원격 호스트로부터의 디스플레이 **사용인증허가 윈도우**(그림 4)가 나타나게 된다. 이때 사용이 허가되면 화면에 연결종료 아이콘(그림 5)이 생성되고, 동시에 원격 호스트에서 실행시킨 X 응용프로그램이 나타나게 된다. 이러한 단계를 거쳐 방화벽 시스템을 통한 X 응용프로그램이 xrelay를 이용하여 실행된다. 예를 들어, 원격 호스트에서 xrelay를 실행하고 X 응용프로그램인 ghostview를 실행하면 그림 4의 사용인증허가 윈도우가 나타나고, 허가하면 디스플레이 화면에 그림 5의 **연결종료 아이콘**과 실행한 ghostview화면이 나타난다. 그림 6은 원격 호스트에서 실행한 X 응용프로그램인 ghostview로 postscript 파일을 본 화면이다

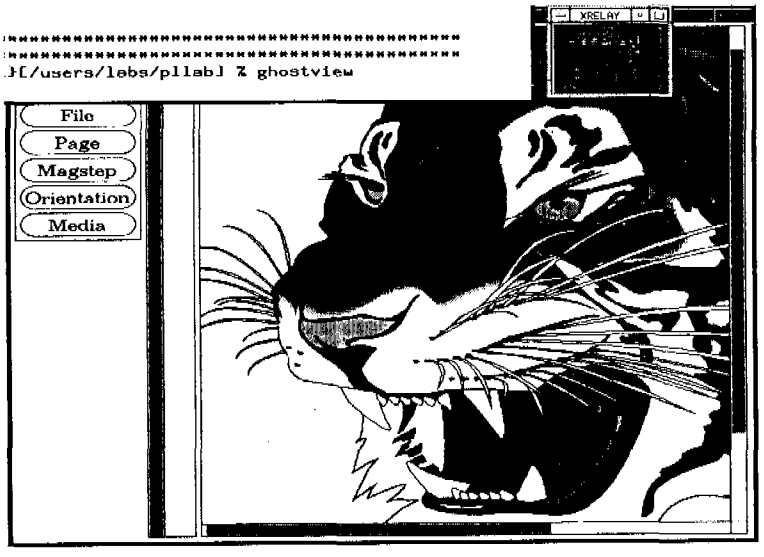


그림 6. X 응용프로그램 실행예

xrelay 서비스를 종료하는 방법에는 세 가지가 있다. 하나는 원격 호스트에서 쉘을 빠져나가는 exit명령사용이다. 더이상 클라이언트가 동작하지 않으므로 서버도 종료하게 된다. 또 하나는 디스플레이 화면상의 연결종료 아이콘을 클릭하는 방법이다. 이때 서버 프로세스는 클라이언트와 디스플레이 양쪽의 연결을 끊고 종료한다. 마지막 방법은 원격 호스트에서 xrelay를 실행할때 timeout 값을 설정(시단위:디폴트 1시간)하는 것이다. timeout 시간만큼 원격 호스트로부터의 X 서비스 요청이 없으면 서버 프로세스는 원격 호스트에서 실행한 모든 X 응용프로그램과 xrelay 클라이언트와의 연결을 끊고 종료하게 된다.

4. 설계 및 구현

4.1 방화벽에서 X 응용프로그램을 지원하기 위한 서버와 클라이언트의 관계

서버 프로그램은 외부 전산망과 내부 전산망 사이에 있는 방화벽에 설치된다. 서버 프로세스는 daemon[4]의 형태로 방화벽 시스템이 기동될 때 동작을 시작하며, 클라이언트로부터 접속 요청이 있으면 자식 프로세스를 생성하여 서비스 해준다. 서버와 클라이언트가 접속하고 자료(data)를 받고 보내는데 사용하는 통신 프로토콜은 *socket*[4]이다. socket은 socket(), bind(), listen(), accept() 등의 c 시스템 함수(c system function)를 이용해 구현되어진다.

먼저 서버 프로세스인 xrelayd 와 클라이언트 프로세스인 xrelay가 어떤 방법으로 방화벽에서 X 응용프로그램을 사용할 수 있게 해주는지 대략적인 흐름을 살펴보겠다.

4.1.1 서버 프로세스인 xrelayd 의 실행 알고리즘

1. socket을 생성하여 클라이언트로부터의 접속을 기다린다.(dsock)
2. 접속을 원하는 클라이언트 중 하나와 접속한다.(connsock)
3. 클라이언트와 접속한 connsocket으로부터 디스플레이와 timeout 값을 읽는다.
4. 디스플레이를 위해 사용가능한 port를 검색한다.(6010 ~ 6100)
5. 디스플레이 자료를 다루기 위한 socket을 생성한다.(lsock)
6. step 4에서 검색한 port number에서 6000을 뺀 값을 step 5 의 lsock을 통해 전송한다.
7. fork() 하여 자식 프로세스를 생성한다.
8. 부모 프로세스는 lsock, connsock을 닫고 새로운 클라이언트의 접속을 기다리기 위해 step 1 의 과정을 반복한다.
9. 자식 프로세스는 step 5에서 생성한 lsock으로 원격 클라이언트의 접속을 기다린다.
10. 클라이언트와 접속이 이루어지면 X 디스플레이 서버 화면에 사용 인증 허가 윈도우를 생성한다. timeout 시간 만큼 접속이 없으면 종료한다.
11. 허가하면 디스플레이와 연결될 socket을 생성(newoutgoing)하고 X 디스플레이 서버에서 연결 종료시 사용할 연결종료 아이콘을 생성한다.
12. step 5의 lsock을 통해 들어온 자료를 step 11의 newoutgoing을 통해 보낸다. Step 12의 과정을 반복한다.

4.1.2 클라이언트 프로세스인 xrelay의 실행 알고리즘

1. argument로 들어온 firewall_name의 호스트와 연결 되는 socket을 생성한다(csock).
2. step 1의 socket을 통해 argument의 display와 timeout 값을 보낸다.
3. step 1의 csock으로 전달될 방화벽 호스트로부터의 응답을 기린다.
4. 응답과 함께 보내진 port_number로 현 프로세스의 환경변수 DISPLAY 값을 **firewall_name: 디스플레이번호**로 지정한다.
5. 원격 호스트의 홈쉘로 전환한다.
   이 이후로는 쉘로서 동작한다.

4.2 서버 프로세스의 구조 및 구현

서버 프로세스인 xrelayd는 방화벽 시스템 내에 daemon의 형태로 동작하는 프로세스로 원격 호스트와 디스플레이 호스트 사이의 연결을 위한 중개자 역할을 하게 된다.

이 프로세스는 그림 7에서와 같이 **포트**(port) 6010을 사용하는 socket을 생성하여 클라

이언트로부터의 요청을 감지하며 대기한다. 클라이언트로부터의 요청이 감지되면 클라이언트로부터 X 디스플레이 서버가 될 디스플레이 값과 timeout 값을 읽고, 자식 프로세스를 생성한다. 부모 프로세스는 계속해서 다른 원격 클라이언트의 접속을 기다리고, 자식 프로세스는 6011 부터 빈 포트를 찾아 클라이언트 프로세스인 'xrelay로 이 포트 number에서 6000을 뺀값을 응답으로 보낸다. X 윈도우 시스템이 포트 6000~6100을 사용하는데 디스플레이 0번이 포트 6000을 사용한다.

그러므로 클라이언트 쪽에서는 6000을 뺀 11에서 100번이 **디스플레이번호**가 되는 것이다. 그런 후 터미널로부터의 모든 입·출력을 막기 위해 file descriptor 0, 1을 close 하고, 클라이언트가 포트 6011~6100에 접속하기를 기다린다. 이 포트로 접속이 감지되면 디스플레이에 이 클라이언트로부터의 요청을 인증하겠느냐는 윈도우를 띄운다. 이때 디스플레이의 사용자가 접속을 허락하면 디스플레이 쪽에 연결종료 아이콘을 생성한다. 이 아이콘은 X 디스플레이 서버 쪽에 예상치 않은 X 응용프로그램이 실행될 경우 또는 더이상의 서비스를 원하지 않을 때 사용자가 연결을 종료할 수 있는 아이콘이다. 이 아이콘은 디스플레이의 호스트가 아닌 방화벽에 있는 서버가 만든 것이기 때문에 이 아이콘 클릭 이벤트에 따라 방화벽의 서버가 실행을 종료한다. 또, 방화벽의 서버 프로세스인 xrelayd가 종료함에 따라 X 디스플레이 서버와 원격 호스트의 클라이언트도 역시 종료하게 된다. 이 아이콘을 생성한 후 클라이언트로부터 받은 디스플레이를 이용하여 디스플레이 호스트와 연결될 socket을 하나 다시 생성한다. 클라이언트와 연결된 socket을 통해 자료를 전송 받고, 받은 자료를 디스플레이와 연결된 socket을 통해 모두 보낸다. 이 부분의 socket을 통한 자료 전송이 방화벽을 통해 X 응용프로그램의 자료가 X 디스플레이 서버로 보내지는 부분이다. 클라이언트 프로세스인 xrelay로부터 timeout 시간만큼 자료를 전송 받지 못하면 클라이언트와 디스플레이 사이의 연결을 끊내고 프로그램은 종료하게 된다. 위에서 설명했듯이 서버의 연결종료 방식에는 두 가지가 있다. 하나는 디스플레이 화면의 연결종료 아이콘을 클릭함으로써 서버 프로세스인 xrelayd를 종료하는 방법이고, 다른 하나는 클라이언트 프로세스인 xrelay로부터 전송 받은 timeout 만큼 클라이언트로부터 자료 전송을 받지 못하면 자동으로 종료하는 방법이다. 클라이언트에서의 연결종료에 대해서는 4.3 에서 설명하겠다.

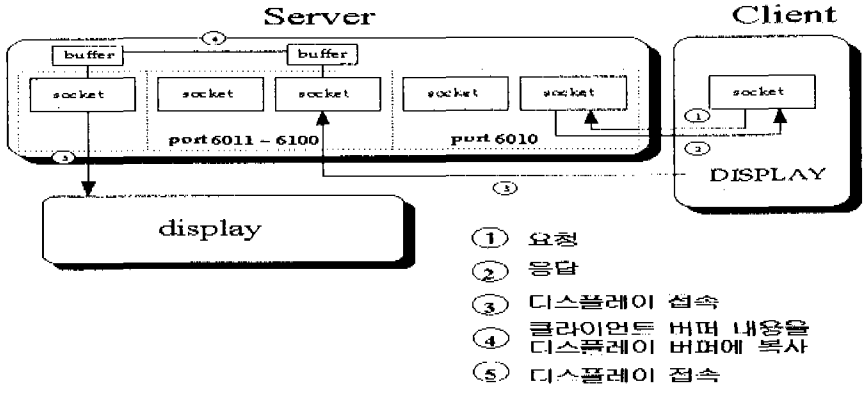


그림 7. Server and Client

4.3 클라이언트 프로세스의 구조 및 구현

서비스를 받을 원격 호스트에서 실행되게 되는 클라이언트는 먼저 argument를 분석하여 어느 방화벽의 서버를 통해 서비스를 받을지를 확인한다. 그런 다음 확인된 방화벽의 포트 6010과 연결될 socket을 생성한다. argument의 display와 timeout 값을 위에서 생성한 socket을 통해 전송한다. 서버로부터의 응답으로 11부터 100 사이의 디스플레이 번호를 전송 받게 된다. 환경변수 DISPLAY를

"firewall_name:디스플레이번호"

로 지정한다. 이 환경변수를 변경한 후 클라이언트 프로세스는 자신의 홈쉘로 바뀐다. 이 쉘에서 X 응용프로그램을 실행하면 서버 프로세스인 xrelayd의 서비스를 받아 방화벽을 통한 X 응용프로그램의 실행이 가능하게 된다. 클라이언트가 서버와의 연결을 종료하려면 쉘 명령어인 exit 명령어를 이용하면 된다.

5. 관련 연구

Digital Equipment Corporation Cambridge Research Lab.에서 구현한 xforward 는 방화벽을 통한 X 응용프로그램 사용을 가능하게 했다[5]. 하지만 사용자가 이 프로그램을 사용하기에는 다음과 같이 여러가지 불편한 면이 있으며, 서비스 측면에서 몇가지 문제점들이 발견 된다.

xforward는 방화벽에 로그인하여 실행하여야 한다. 방화벽에 로그인 ID를 가지고 있어야 할 뿐만 아니라, 서비스를 받으려면 항상 방화벽에서 이 프로그램을 실행해야하는 것이다. 이런 사항을 고려하여 xrelay에서는 방화벽에서 항상 대기하고 있는 daemon형태의 서버 프로세스와 원격 호스트에서 필요할 때마다 서비스를 요청할 수 있는 클라이언트로 구성된다. 그리고 socket 통신에 이용될 포트를 6000 부터 사용하는 점에도 문제가 있다. X 윈도우 시스템은 포트 6000~6100을 사용한다. 그렇기 때문에 방화벽에서 xforward를 실행한 뒤 xinit을 실행하면 포트 6000이 xforward에의해 이미 사용되고 있기 때문에 xinit은 실행되지 않는다. 그렇기 때문에 xrelay에서는 다중화면(multiscreen)을 고려하여 포트 6010을 서버와 클라이언트가 접속하는데 이용하고, 포트 6011~6100을 디스플레이를 위해 사용한다. 또, xforward는 xforward를 실행할때 허용한 호스트만이 서비스를 받을 수 있다. 그런데 허용된 호스트에서 자신이 아닌 다른 사용자가 방화벽의 디스플레이를 알아내어 이 디스플레이로 접근할 시 방화벽의 xforward를 kill하는 방법밖에 다른 사용자의 불법적인 접근으로부터 벗어날 방법이 없다. 이 문제를 해결하기 위해 xrelay에서는 디스플레이 화면에 연결종료 아이콘을 생성하여 실행한 X 응용프로그램 한번에 연결을 종료할 수 있도록 했다.

TIS(Trusted Information Systems) Firewall Toolkit에서도 방화벽을 통해 X 응용프로그램을 지원하기 위한 프로그램을 제공한다[6]. x-gw 가 바로 그것인데 xrelay와 비슷하게 서버 daemon이 클라이언트에 서비스해주는 방법을 취한다. 그런데 이 프로그램은 tn-gw나 rlogin-gw의 기반하에 제공되므로 TIS Toolkit을 설치해야만 서비스를 이용할 수 있다는 단점이 있다.

6. 결론 및 추후 연구 방향

철저한 보안의 필요성에의해 개발된 대부분의 방화벽 시스템에서는 X 응용프로그램을 방화벽의 내부디스플레이에 실행할 수 있도록 지원하지 않는다. 그런데, 서버인 xrelayd를 방화벽에 설치하고 xrelay 클라이언트를 원격 호스트에 설치하여 사용함으로써 방화벽을 통해 X 응용프로그램을 방화벽 내부의 디스플레이에 실행할 수 있게 되었다. 이렇게 xrelay 서버와 클라이언트를 이용하여 방화벽으로 인해 장애가 되었던 외부 전산망과 내부 전산망 사이의 X 응용프로그램 실행을 효과적으로 지원하게 되었다. 이 연구 과정에서 내부 전산망과 외부 전산망 사이에 위치하는 인터넷상의 게이트웨이 호스트(munsu.ulsan.ac.kr)를 방화벽으로 가정하고, 원격 호스트인 인터넷상의 시스템공학연구소(SERI)의 Cray에서 xrelay 클라이언트를 실행하여 내부 전산망의 디스플레이에 X 응용프로그램 실행을 실험하였다. 현재 xrelay는 Solaris 2.4, SunOS 4.1에서 실험해 보았다.

앞으로 xrelay를 다음과 같이 발전시켜 나아갈 계획이다. 클라이언트의 요청이 있을때 서버가 자식프로세스를 생성하여 서비스하는 현재의 방법을 개선하여 IEEE의 표준인 POSIX Threads로 서비스할 수 있게 구현하려 한다. 그리고, Microsoft Windows용 클라이언트를 winsock을 이용하여 구현함으로써 PC시스템에서도 xrelay 서비스를 받을 수 있게 할 예정이다.

참 고 문 헌

1.. William R. Cheswick and Steven M. Bellovin, Firewalls and Internet Security & Firewall, Addison-Wesley, 1994
2. Herve Schauer and Christophe Wolfugel, "In UNIX sevurit Symposium III Proceedings", p.49-61, USENIX, 1992
3. Marcus J. Ranum and Frederick M. Avolio, "A Toolkit and Methods for Internet Firewalls", Trusted Information Systems, Inc., 1994
4. W. Richard Stevens, Advanced Programming in the UNIX Environment, Addison-Wesley, 1992
5. G. Winfield Treese and Alec Wolman, "X Through the Firewall, and Other Application Relays", DEC CRL Tech. Rep., 1993
6. "TIS Firewall Toolkit", Trusted Information Systems, Inc., 1994
7. Douglas E. Comer and David L. Stevens, Internet Working with TCP/IP, Prentice-Hall, 1993
8. W. Richard Stevens, UNIX Network Programming, Prentice Hall, 1991
9. 오익균, 정윤종, "인터넷 보안과 방화벽", KRNET'95, p.475~580, 1995