Linux 상에서의 Network Streaming Service 설계

김영만*, 박홍재*, 허성진**, 최완**
*국민대학교 컴퓨터학부 통신실험실, **한국전자통신연구원
ymkim@cclab.kookmin.ac.kr, elitebasic3@hotmail.com, sjheo@etri.re.kr, wchoi@etri.re.kr

Design of the Network Streaming Service based on Linux

Young Man Kim*, Hong Jae Park*, Seong Jin Heo**, Wan Choi**
*Communication Lab, School of Computer Science, Kookmin University,
**Electronics and Telecommunications Research Institute

요 약

본 논문에서는 개방형 운영체제인 리눅스상에서 package streaming service를 제공하는 Network Streaming Service(통칭 NSS)를 설계하는 문제를 다룬다. NSS는 client가 server측에 설치된 소프트웨어 package를 마치 client 자신의 local disk상에 이미 인스톨된 패키지를 실행하는 것처럼 동작하도록 해준다. NSS는 client측의 NSS Mounter, Package Streaming Consumer, NSS Configurator와 server측의 NSS Load Balancer, Package Streaming Supplier, NSS Manager, Accounting & Security Manager로 구성되며 local disk caching과 background prefetching 기능을 사용하여 향상된 서비스를 제공한다.

1. 서 론

기존 리눅스 기반에서 동작하는 파일 서비스중의 하나인 NFS(Network File System)는 컴퓨터 사용자가 원격지 컴퓨터에 있는 파일을 자신의 하드 디스크에 있는 것처럼 마운트 해서 검색, 저장 및 수정을 가능하게 해주는 client/server형 네트워크 파일자원 공유 서비스를 말한다. NFS의 기능[1][2]은 client측의 nfs 모듈과 server측의 nfsd 데몬에 의하여 제공된다. NFS를 통하여 네트워크에 연결된 컴퓨터들에게 파일시스템을 제공한다는 아이디어를 한단계 향상시킨 것이 NSS(Network Streaming System)이다. 최근 정착된 초고속망에 연결된 모든 가입자들에게 방대한 양의 소프트웨어 패키지를 설치과정을 생략한 채 실시간으로 실행시킬 수 있도록 하는 서비스가 NSS이며 NSS 서비스가 가능한 형태로 재배치된 소프트웨어 패키지가 클라이언트의 요구에 응하여 스트리밍형태로 NSS서버측에서 흘러와서 즉시 실행 사용가능하도록 해준다. 국내 소프트온넷사의 Z!stream[3]이 SW streaming기술로 이미 알려져 있는데 현재 windows 환경상에서만 동작하고 있다. NFS는 RPC를 통해서 읽어온 page를 메모리의 page cache에 저장하게 된다. 하지만 page cache의 용량은 매우 제한적이여서 네트워크로 읽었던 page를 다시 읽어와야 하는 경우가 자주 발생하며, 이러한 문제점은 서버에게 부담이 될 뿐만 아니라 네트워크의 트래픽을 증가시키게 된다. 그러나 NSS는 RPC를 통해 한번 읽은 page를 page cache뿐만 아니라 client의 local disk에도 저장하여 이후의 모든 참조에 즉각적인 응답을 주게되어 네트워크 지연시간을 생략함으로써 NFS보다 실행성능을 개선시킬 수 있다. 또한 background prefetching 기법을 통해 local disk cache file의 적중률을 높일 수 있다. 본 논문에서는 이러한 기법들을 포함하는 NSS의 설계에 관하여 설명하고자 한다.

2. NSS 설계

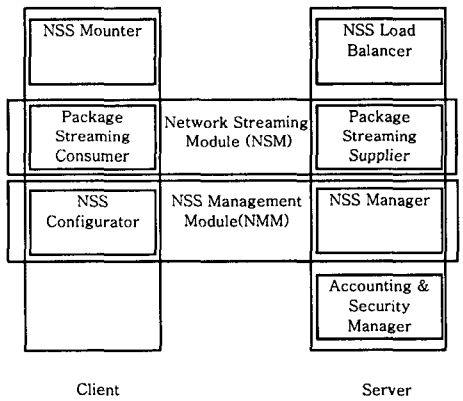

그림 1 NSS 구조도

NSS는 client가 네트워크의 streaming service를 통하여 server측의 package를 자신의 local disk에 이미 설치되어 있는 것처럼 즉시 사용할 수 있도록 해주는 시스템이다. [그림 1]은 NSS server와 client의 내부구조를 보여주는데 실질적인 streaming service는 NSM(network streaming module)과 NMM(NSS management module)에서 이루어진다.

2. 1 Server 측 소프트웨어의 구성 및 역할

2.1.1 NSS Load Balancer

여러 server를 사용할 경우 client의 요청이 한 server에 집중되는 것을 막기위해 client의 요청을 각 server에 분산시키는 역할을 한다. 이로 인해 client들은 신속한 streaming service를 받아 package의 실행속도를 높일 수 있다.

2.1.2 PSS(Package Streaming Supplier)

PSS는 server측에 설치된 package들의 집합을 말한다. NSM은 client가 요구한 page를 PSS에서 가져와 client에게 제공한다. NSM이 PSS에서 page를 가져오기 위해서는 NSS Mounter에 의해서 client와 server가 연결되어 있어야만 한다. 이러한 연결을 위해서 PSS와 관련된 모든 파일은 NSS Mounter에 의해서 연결된 server측 패키지 디렉토리의 하위에 모두 존재해야만 한다. 예를 들면 [그림 2]에서는 오피스 관련 오픈 소프트웨어 패키지인 OpenOffice 파일들이 디렉토리 '/usr/lib/openoffice' 안에 모두 모여 있는 것을 보여준다.

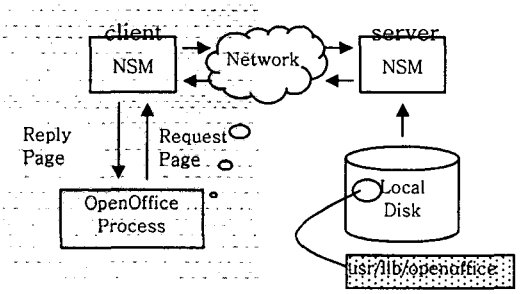

그림 2 Package Streaming Supplier의 예

2.1.3 NSS Manager

Client가 server에게 streaming service를 요청하면 server의 NSS Manager는 효율적인 prefetching을 실행할 수 있도록 하기위해 준비해온 priority table을 client로 보낸다. 이 priority table은 client가 사용빈도가 높은 page를 우선적으로 streaming 받기 위한 정보를 포함하고 있고, 이에 따라 client는 server에 prefetching 할 page를 요구하게 된다. NSS Manager는 client에서 page를 요구할 때마다 priority table의 해당 page의 priority value를 높여줌으로써 priority table이 업데이트된 정보로 구성되도록 관리한다.

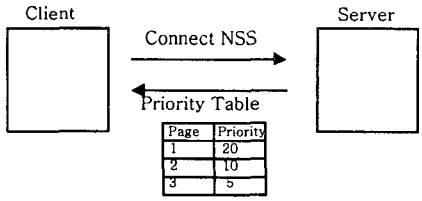

그림 3 NSS Manager의 역할

[그림 3]에서와 같이 client가 NSS Mounter에 의해서 server에 접속 될 때 server에서는 응답메시지에 NSS Manager가 관리하는 priority table을 포함한 접속정보를 client에 보낸다. Priority table은 이후에 client가 prefetching할 파일과 page를 선택하는 기준으로 사용된다. 이렇게 prefetching할 page에 대한 선택을 client에게 담당시킴으로써 server의 부하를 줄일 수 있다. 또한 client가 server에 page를 요구할 때 마다 해당 page의 priority value를 증가시키는데, client는 server접속 시 제공받은 priority table을 참조하여 priority value가 높은 page부터 server로부터 prefetching 한다.

2.1.4 Accounting & Security Manager

사용자 계정 및 보안과 관련된 작업을 한다. 이는 server에 접속을 시도한 여러 client들 중 인증 받은 사용자만이 streaming service를 받도록 하기 위함이다. 관리자는 client account 할당 및 관리를 효율적으로 제공한다.

2. 2 Client측 소프트웨어의 구성 및 역할

2.2.1 NSS Mounter

NSS Mounter는 client가 network streaming service를 받기 위하여 server와의 연결을 초기화할 때 사용하는 module이다.

[그림4]는 NSS Mounter의 동작과정을 나타낸 것이다. Server에 mount demon이 실행되어 있는 상태에서 client는 server의 mount demon port number를 요청한다. client는 받은 port number로 통신을 하며 RPC call을 이용하여 server의 super block 을 가져오게 된다. 이렇게 함으로써 client는 자신의 local disk에 설치되어 있지 않은 server의 package를 실행할 수 있게 된다.

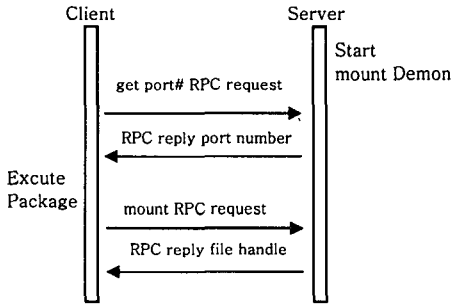

그림 4 NSS Mounter 동작과정

2.2.2 Package Streaming Consumer

Server로부터 package streaming을 받기 위해 데이터의 송신을 요구하는 부분이다. [그림 5]는 client가 프로그램을 실행하기 위해 필요한 page를 NSM에 요구하면 server측의 PSS가 해당 page를 client에 streaming 해주는 것을 나타낸다. 이때 server로부터 받은 data는 client의 disk cache에 저장함으로써 이후의 모든 참조에 대해 네트워크를 통할 필요없이 local disk cache로부터 해당 page가 로드된다.

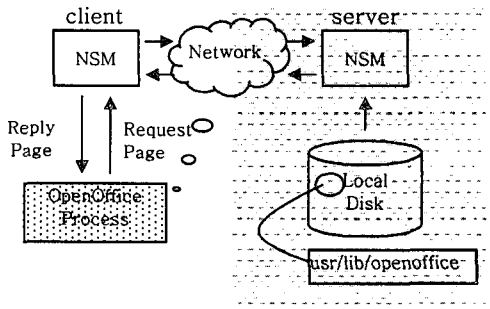

그림 5 Package Streaming Consumer의 예

2.2.3 NSS Configurator

Client에 관한 정보를 관리하기 위해 server측과 통신하는 부분이다. 이때 server는 prefetching을 하기위한 priority table을 client로 보낸다. 이 priority table은 서비스 받을 server측 package의 file 및 page의 우선순위 정보를 포함하고 있고 이를 받은 client는 priority table에 따라

prefetching을 하게 된다.

3. NSS 주요특징

NSS가 NFS와 구분되는 주요 특징은 첫째, NSS에서는 패키지 내의 모든 파일은 완성된 상태에 있으며 client로부터의 수정이나 생성, 삭제를 허용하지 않는다. 한편 NFS는 파일들의 업데이트 상황을 유지하기 위하여 수 많은 RPC메시지들을 주기적으로 혹은 요구에 응하여 송수신 하도록 규정되어 있다. 둘째, NSS는 액세스 성능을 향상시키기 위하여 streaming된 블록들을 local disk 에 영구 보존시킴으로써 향후에도 계속 사용할 수 있도록 하지만 NFS는 이를 금지하고 있다. 셋째는, NSS에서 보다 신속한 패키지 실행을 위하여 prefetching 기능을 추가한 것이다.

3. 1. Local disk cache

기존의 NFS는 RPC를 통해서 읽어온 page를 메모리의 page cache에 저장하게 된다. 하지만 page cache의 용량은 매우 제한적이다. 그래서 네트워크로 읽었던 page를 다시 읽어와야 하는 경우가 자주 발생하며, 이러한 문제점은 서버에게 부담이 될 뿐만 아니라 네트워크의 트래픽을 증가시켜서 성능 저하의 원인이 된다. NSS는 RPC를 통해 한번 읽은 page를 page cache뿐만 아니라 client의 local disk에도 저장함으로써 이후 같은 page를 사용할 때 네트워크를 통해서가 아닌 local disk의 cache file을 사용하여 NFS보다 성능을 개선시킬 수 있다[4]. 이 서비스는 전원이 나가거나 프로그램을 재실행 할 때에도 지속되도록 한 고효율, 고신뢰도의 서비스이다.

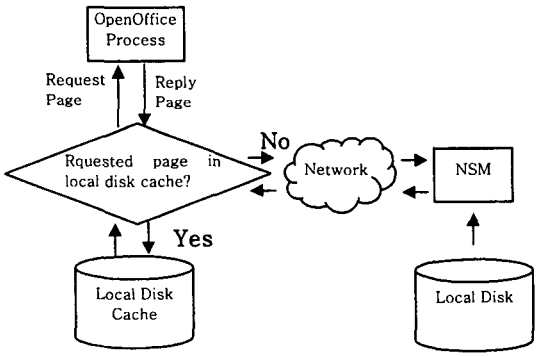

그림 6 local disk cache 기본흐름

Package Streaming Consumer에 의해서 일부분이라도 사용되어진 파일은 client의 local disk cache의 cache file내에 생성되어 지고, 생성된 cache file의 block 정보를 관리하기 위해서 별도의 disk공간에 bitmap이 만들어진다. 이 bitmap은 package streaming consumer가 사용하길 원하는 page가 local disk cache에 존재하는지 판단하는데 사용된다. 모든 cache file은 client가 NSS Mounter를 통해서 server와 연결할 때 옵션으로 주어진 local disk cache의 루트 디렉토리에 저장되며 cache file의 이름은 "host-name"+ "device-number"+ "inode-number"의 문자열로 이루어져 cache file의 관리가 용이하도록 한다.

3. 2. Background prefetching

Client가 요구하는 page만을 전송하는 것이 아니라 client가 다음에 어떤 page를 사용할지 예측해서 client가 요구하기 전에 미리 cache file에 저장한다면 cache file의 적중률을 높일 수 있고 client는 좀더 빠른 속도로 package를 실행할 수 있을 것이다. 이때 어떤 page를 prefetching할지를 결정하는 기준이 되는 것은 client가 초기 연결 시에 server로부터 받은 priority table이다.

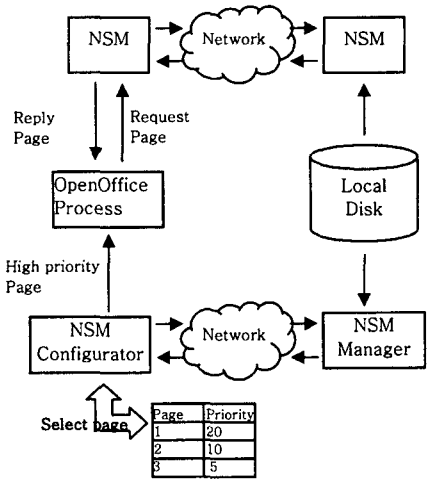

그림 7 background pre-fetching

[그림 7]은 background prefetching에 대한 흐름도인데 client가 당장 필요한 page를 server에 요청하는 것과는 별개로 priority table을 기준으로 하여, 사용 빈도가 높은 page를 background로 prefetching하여 local disk의 cache에 저장하게 된다.

4. 결론 및 향후 연구 방향

본 논문에서는 리눅스의 NFS모듈을 참고하여 client가 server측에 설치된 package를 network의 streaming service를 통하여 이용 가능하게 하는 NSS에 대한 구조설계와 그 주요특징들에 대하여 설명 하였다. NSS는 NFS상에 local disk caching과 background prefetching기능을 추가하였기 때문에 client가 server의 package를 실행할 경우 높은 실행속도를 보장한다. 현재 최적화된 streaming service를 구현하기 위하여 prefetching에 관한 효율적인 알고리즘을 연구하고 있으며 또한 NSS기능의 일부를 구현중에 있고 나머지도 조만간 구현 완료 한후 성능평가에 들어갈 예정이다.

참 고 문 헌

[1] "리눅스 커널 분석 2.4", 박장수 저 가메출판사, 2003.
[2] http://nfs.sourceforge.net.
[3] "Z!stream White Paper", http://www.softonnet.com.
[4] "An Enhanced Disk-Cashing NFS Implementation for Linux" , http://www.cs.washington.edu/homes/gjb/doc/enhanced-linux-nfs-client/index.html.