분산 환경의 대규모 클러스터를 관리하기 위한 RISE 시스템의 설계 및 구현

(The Design and Implementation of RISE for Managing a Large Scale Cluster in Distributed Environment)

박 두 식 † 양 우 진 ††† 반 민 호 †† 정 갑 주 †††† 이 종 현 †††††
(Doosik Park) (Woojin Yang) (Minho Ban) (Karpjoo Jeong) (Jonghyun Lee)

이 상 문 ††† 이 창 성 †† 신 순 철 ††† 이 인 호 ††
(Sangmoon Lee) (Changsung Lee) (Soonchurl Shin) (Inho Lee)

요 약 본 논문에서는 지리적으로 분산되어 있는 클러스터 시스템자원들을 효율적으로 활용하기 위한 3-tier 구조의 원격 설치 및 백업 방안을 소개한다. 최근에는 클러스터 시스템이 수백 노드 이상의 대규모 시스템이며, 공인망과 사설망이 혼재되는 복잡한 네트워크 환경으로 구성되고 있다. 따라서, 대규모 클러스터 시스템에 적합한 클러스터의 OS 설치와 원격지에서 클러스터 노드의 장애를 효과적으로 복구하는 것이 중요하다. 하지만 기존의 2-tier 구조의 클러스터 설치 및 이미지 백업 방법들은 공인망과 사설망으로 구성되어 있는 클러스터의 경우, 원격지에서 접근과 관리가 어렵다. 이러한 문제점을 해결하고자 본 논문에서는 3-tier 구조의 RISE(Remote Installation Service Environment) 시스템을 제안하고자 한다. RISE 시스템은 2-tier 구조의 마스터 노드 역할을 관리노드(GRISE)와 지역관리노드(LRISE)로 나누어 줌으로써 다양한 네트워크환경하에서 초기설치 및 장애 발생시 효과적으로 지원할 수 있으며, 관리노드와 지역관리노드들의 동기화 기능을 통해 지역관리노드들의 안정성을 보장하고 있다.

64개 노드의 클러스터 시스템과 Gigabit 네트워크 시스템을 활용한 실험을 통하여, 1.86 GByte의 시스템 이미지를 5분 53초 안에 확보 할 수 있었고, 64개 노드 클러스터 시스템의 초기설치 작업을 평균 17분 38초 안에 완료할 수 있었다.

키워드 : RISE, 대규모 클러스터, 3-tier 구조

Abstract In this paper, the way of remote installation and back-up of 3-tier structure is introduced for efficient utilizing the cluster system resources distributed at several places. Recently, cluster system is constructed as the system of over hundreds nodes under complex network system mixed with public networks and private networks. Therefore, the OS installation method suitable for the large scale cluster system and the remote recovery of failure nodes are important. However the previous researches which are based on 2-tier architecture may not provide the efficient cluster installation and image back-up method when the network of cluster system is composed of several private networks and public networks. In this paper, RISE (Remote Installation Service and Environment) based on the 3-tier architecture is proposed to solve this problem. In our approach, the managing node's role is divided into the global master node (GRISE) and the local master node (LRISE) to provide the efficient initial system deployment and remote failure recovery of distributed cluster system under the various network systems. Also, LRISE's availability is ensured under the complex network environments by adopting the auto-synchronization mechanism between GRISE and

† 정 회 원 : 삼성전자 컴퓨터서버팀
doos@samsung.com
†† 비 회 원 : 삼성전자 컴퓨터서버팀
mhban@samsung.com
cs1023.lee@samsung.com
inholee@samsung.com
††† 비 회 원 : 삼성종합기술원
woojin.yang@samsung.com
sm64.lee@samsung.com
scshino@samsung.com
†††† 종신회원 : BK21 u-Science 기반 신기술융합 사업단 단장
jeongk@konkuk.ac.kr
††††† 학생회원 : 건국대학교 컴퓨터공학과
lejohy@gcslab.konkuk.ac.kr
논문접수 : 2005년 9월 5일
심사완료 : 2006년 4월 7일

LRISE. In this work, a 64-node cluster system with gigabit network system is utilized for the experiment. From the experimental result, the system image with 1.86GB data can be obtained in 5 minutes and 53 seconds and the image-based installation of 64-node system can be carried out in 17 minutes and 53 seconds.

Key words : RISE, large scale cluster, 3-tier architecture

1. 서 론

고성능계산을 포함한 대규모 연산처리 분야에서 클러스터 시스템은 하나의 표준 시스템으로서 자리매김하고 있고 최근에 와서 클러스터 시스템의 크기 및 구성이 매우 복잡해지고 있는 추세에 있다. 국내의 경우에도 수백 CPU 이상의 시스템들이 구축되고 있고, 전 세계적으로는 수천 CPU 이상의 시스템이 구축되고 있다[1]. 대규모 클러스터의 경우 서브 시스템들이 공인망과 사설망이 혼합된 다양한 네트워크를 통해 연결되어 있으며, 이러한 서브 시스템들이 지리적으로 분산되어 있는 경우도 있다. 그런데 클러스터의 지리적 분산과 노드 수의 증가는 관리비용의 증가를 가져왔으며, 방화벽과 복잡한 주소체계 등으로 원격지에서의 사설망 내의 클러스터 노드들에 대한 관리를 더욱 어렵게 만들었다. 따라서 대규모 분산 클러스터의 효율적인 관리방법이 요구되고 있으며, 특히 클러스터 시스템의 초기 설치 및 설정, 시스템 운영 중 클러스터 노드 장애 발생(운영체제 오작동, 저장장치 오류 등)에 대처 방안, 운영자의 특정 필요(대상 노드의 용도 전환 등)에 따른 시스템의 재설치 등의 효과적인 방법 등이 필요하다. 하지만 기존에 구현되었던 시스템[2~5]들은 다양한 네트워크에 분산되어 있는 클러스터 환경에 대한 고려가 미비했다.

본 연구에서는 이러한 대규모 분산 클러스터 환경에 적합한 원격 통합 설치 및 이미지 백업 시스템인 RISE (Remote Installation Service Environment)를 제안하고자 한다. RISE의 3-tier아키텍처를 통해 다양한 네트워크 환경에 분산되어 있는 클러스터를 효과적으로 관리하고 대규모 클러스터의 안정성을 높일 수 있는 방안을 제시 하였다.

2. 분산 클러스터 환경과 원격 시스템 이미지 관리

그림 1에 나타난 바와 같이, 각각의 클러스터는 지리적으로 분산되어 있으며 하드웨어 스펙 및 환경, 용도가 다르고, 각 클러스터들은 한 사람 또는 몇 사람의 관리자가 시스템을 나눠 운영하고 있다. 이러한 환경 하에서 전체 시스템에 대한 체계적인 관리 및 효율적인 활용은 매우 어렵다. 또한 장애 발생 시 노드의 OS재설치, 응용 프로그램 설치 및 환경 설정은 시스템의 생산성 및

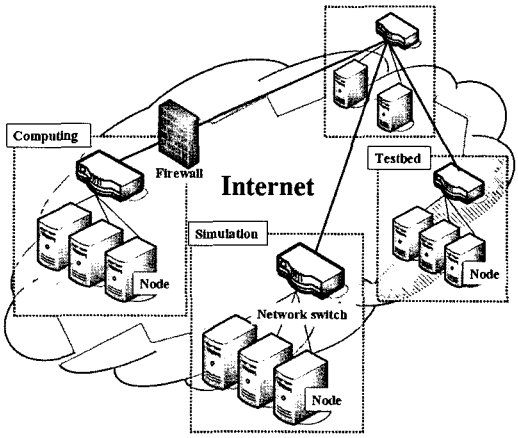

그림 1 분산 클러스터 환경

안정성을 매우 떨어뜨린다. 이를 방지하기 위해 원격지상에서 각 클러스터 노드의 시스템 이미지에 대한 효율적 관리(백업, 설치)가 무엇보다 필요하다.

통상적인 시스템 이미지의 정의는 시스템이 목적된 역할을 정상적으로 수행하기 위한 OS, 각종 응용프로그램, 시스템 환경설정 파일 및 필요에 따라 사용자 데이타 영역 등을 포함하는 이미지 파일과 관련된 다양한 속성을 포함하는 메타 데이타를 의미한다. 이와 같은 시스템 이미지의 백업 또는 설치를 위한 다양한 연구 및 개발이 지속적으로 수행되어오고 있는데, 관련된 중요 연구로서, RPM(Redhat Package Manager)에 기반을 둔 Linux 자동화 설치 방법인 KickStart[2]를 예로 들 수 있다. KickStart는 운영환경을 설치하고자 하는 시스템상에 Install Agent를 가동하고 이 Agent가 사용자에 의해 정의된 명세파일에 따라 지정된 RPM Package를 순차적으로 설치하여 시스템을 구성하는 방식이다. 그러나 KickStart는 다수 Node의 설치를 일괄적으로 할 수 없고 적용 대상이 각각의 단위 Node로 제한된다는 한계가 있다. 이와 같은 단점을 극복한 Tool로서 OSCAR (Open Source Cluster Application Resources)[3]와 NPACI-ROCKS[4]등을 예로 들 수 있다. OSCAR 와 NPACI-ROCKS는 KickStart와 동일하게 RPM방식의 설치 방법을 채택하고 있으나, 무엇보다도 다수 Node의 설치를 관리할 수 있는 통합적인 User Interface가 제공된다는 점에서 그 차별성이 있다. 이와 같은Kick-

Start, OSCAR 및 NPACI-ROCKS는 일단 설치하고자 하는 환경이 사전에 명확히 정의되어 있고, 해당 Software Package가 완벽히 준비되어 있다면, 최초의 시스템 환경 구성에는 비교적 손쉽게 사용할 수 있다는 장점이 있다. 그러나 설치 이외에 백업기능이 없기에 현재 사용 중인 상태 그대로의 시스템 이미지 백업이 불가능하다. 이외에도 System Installation Suite[5]의 경우는 파일 시스템을 직접 접근하여 현재 상태의 시스템 이미지를 파일 기반으로 백업 및 설치가 가능하다. 이와 같이 언급된 다양한 클러스터 설치 및 이미지 백업 Tool에 대한 각각의 특징은 표 1과 같다.

표 1 클러스터 설치 및 이미지 백업 Tool

	설치	백업	지원 Platform	다수 노드 통합 관리
Kick Start	O	X	RPM based Linux	X
OSCAR	O	X	RPM based Linux	O
NPACI-ROCKS	O	X	RPM based Linux	O
System Installation suite	O	O	Linux	O

이들 연구는 모두 하나의 마스터 노드에서 이미지 관리 기능을 수행하는 2-tier 구조를 택하고 있다. 2-tier 구조에서 가장 큰 문제점은 하나의 관리 노드에 장애가 발생하면 전체 시스템의 동작을 보장 받을 수 없다는 것이다. 심각한 하드웨어 장애로 인해 마스터 노드의 정보가 분실되었을 때 복구할 방도가 없다. 또한 수백에서 수천 개의 노드로 구성된 시스템에서 하나의 마스터 노드가 다른 모든 노드를 관리하는 것은 효율성이 크게 떨어진다. 분산 네트워크 환경에서 클러스터 노드가 글로벌 주소 체계를 사용하지 않는 경우 원격지 관리는 더욱 어렵다. 이들 문제점을 효과적으로 해결하고자 본 논문에서는 RISE를 제안하고자 한다.

3. RISE(Remote Installation Service Environment)

3.1 RISE의 개요

RISE는 원격지에서 대규모 클러스터를 통합 설치하기 위한 웹기반의 통합환경을 관리자에게 제공한다. 그림 2에서 알 수 있듯이, RISE는 GRISE(Global RISE), LRISE(Local RISE), NRISE(Node RISE), WRISE (Web RISE)로 구성되어 있으며, 각각의 기능은 아래와 같다.

- GRISE(Global RISE) : LRISE로부터 생성된 시스템 이미지 정보(이미지 메타데이타와 이미지 파일)를 주기적으로 복제하는 백업 이미지 서버이며, LRISE는 장애 시에 GRISE로부터 이미지 정보를 전송 받아 자동으로 이미지 파일을 복구할 수 있다. 또한 GRISE

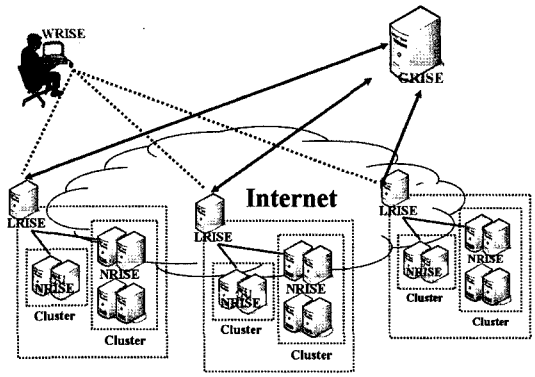

그림 2 RISE 시스템의 개요

는 각 서버 간 주소 및 설정 정보를 인덱스 테이블에 저장하고 있으며 주기적으로 업데이트를 수행한다. 각 서버는 설정변경을 주기적으로 체크하는 쓰레드를 통해 서버 간 자동 설정 변경 기능을 수행한다.

- LRISE(Local RISE) : 클러스터 내의 각 노드에 대한 원격 백업/설치의 제어 및 관리, 모니터링 기능을 수행한다. 또한 클러스터 노드의 백업/설치 시 필요한 이미지 저장소의 기능도 제공한다. 즉, WRISE로부터 요청되는 작업을 NRISE로 넘겨주며 NRISE의 정보를 WRISE로 전달해주는 역할을 담당한다.
- NRISE(Node RISE) : 노드 상에서 이미지의 백업/설치를 담당하는 에이전트로 LRISE와의 통신 및 이미지 백업/설치 도구의 래퍼로 이루어져 있다. NRISE는 플러그인 설계를 통해 이미지 백업/설치 도구와 상관없이 이미지 백업 방법의 확장성을 제공한다.
- WRISE(Web RISE) : 웹 기반 사용자 인터페이스로 이미지 서버 관리 기능, 이미지 관리 기능(원격 백업/설치 및 이미지 서버간 복사, 이동, 삭제)과 노드 관리 기능(클러스터 초기 노드의 검색 및 추가, 삭제), 클러스터 별 관리자 접근 권한 관리, GRISE 관리 기능을 제공한다.

3.2 3-Tier 구조

기존의 2-tier구조의 시스템은 공인망과 사설망이 혼재되어 있는 경우에는 클러스터를 효율적으로 관리할 수 없다. 본 연구에서 제안한 RISE는 그림 3과 같이, 기존의 관리 서버(GRISE)와 관리 노드 사이에 지역 관리 서버(LRISE)를 둔 3-tier 구조를 채택함으로써 다양한 네트워크를 효과적으로 지원할 수 있다.

RISE에서는 기존의 클러스터 관리 노드의 역할을 LRISE와 GRISE에게 분산시켰기 때문에, LRISE는 로컬 네트워크 내의 클러스터 노드만을 관리하고 GRISE는 이들 LRISE만을 관리하는 역할을 수행한다. 이러한 구조는 GRISE가 로컬 네트워크 내의 노드와 직접 통신

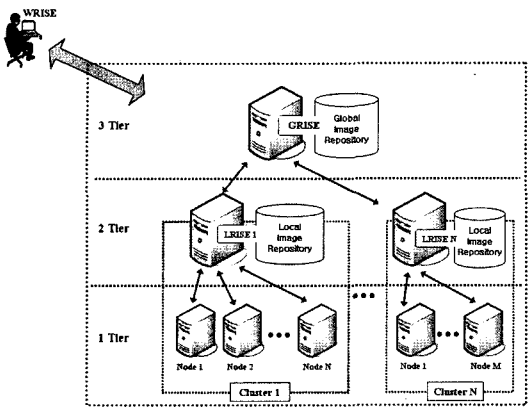

그림 3 RISE의 3-tier 구조

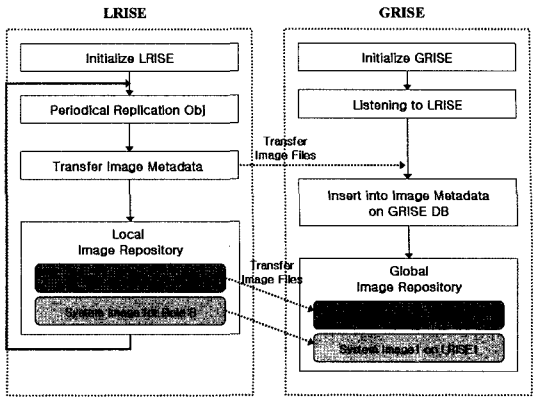

그림 4 LRISE로부터 GRISE 이미지 자동 복제 과정

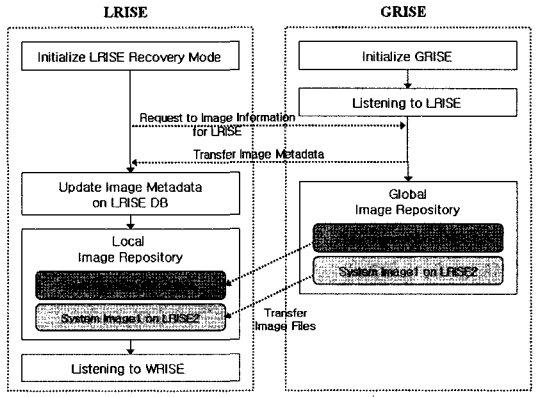

그림 5 GRISE로부터 LRISE 이미지 자동 복구 과정

할 필요가 없으므로, 클러스터 내의 노드가 사설망으로 구성되어 있다고 하더라도 하나의 노드가 외부 인터페이스를 가진다면 모든 노드가 RISE의 관리를 받을 수 있게 된다. RISE는 이러한 3-tier 구조를 제안함으로써 다양한 네트워크 망에 분산되어 있는 클러스터를 효과적으로 지원할 수 있고 확장성도 보장 받을 수 있다.

3.3 이미지 자동 복제 및 장애 복구

앞에서 언급한 것처럼, LRISE는 자동으로 클러스터 이미지를 GRISE에 복제하고, 만약 LRISE에 장애가 발생하면 GRISE로부터 이미지 메타 데이타와 이미지 파일을 자동 복구한다. LRISE는 복제 쓰레드를 통해 주기적으로 LRISE의 시스템 이미지 정보(이미지 메타데이타와 이미지 파일)을 GRISE로 복제하고 장애 시 GRISE는 LRISE-ID를 통해 LRISE를 구분하여 복구한다. 이러한 과정을 통해서 RISE 시스템의 안정성을 증가 시켰다. 그림 4에 나타난, LRISE로부터 GRISE의 이미지 자동복제 과정을 살펴보면 다음과 같다.

① LRISE는 서버 시작 시 주기적으로 GRISE에 접속하여 현재 저장하고 있는 이미지의 복제 여부를 확인한다.

② 복제할 이미지가 존재하면 해당 이미지의 정보를 GRISE로 전송하고 LRISE의 복제여부 플래그를 DONE으로 변경한다. 복제의 실패 시 플래그를 FAIL로 바꾸고 다음 주기에 다시 복제를 시도한다.

③ 복제할 이미지가 존재하지 않는다면 일정 주기 동안 대기한다.

LRISE의 장애(저장 장치 오류, 운영체제 오류, 해킹 등) 발생 시 클러스터 관리자는 LRISE 복구 모드를 통해 기존 LRISE의 상태를 복구한다. 그림 5에 나타난, GRISE로부터 LRISE 서버의 이미지 자동 복구 과정을 살펴보면 다음과 같다.

① 장애 발생 시 관리자는 LRISE를 복구모드로 실행한다. 이때 LRISE-ID는 복구의 ID로서 사용된다.

② LRISE는 LRISE-ID를 통해 GRISE에 복구를 요청하고 GRISE는 LRISE-ID에 대한 시스템 이미지 정보(이미지 메타데이타와 이미지 파일)를 검색하여 LRISE로 전송한다. 이때 LRISE의 복제 여부 플래그는 이미 GRISE상에 이미지가 존재하므로 항상 DONE으로 표시한다. 데이타의 전송 중 실패가 발생하면 3번의 재시도를 통해 재전송한다.

③ 복제가 완료되면 LRISE는 WRISE의 요청을 처리하기 위해 대기한다.

이러한 과정을 통해 LRISE의 이미지가 GRISE로 백업되고 이후 LRISE 장애 발생시 GRISE는 LRISE의 복구를 수행한다. GRISE와 LRISE의 복구 과정은 기존의 2-tier 구조에서 관리 노드의 장애 발생 시 서비스를 계속 제공하기 어려웠던 문제점을 해결할 수 있다.

3.4 자동 설정 변경 동기화

RISE 시스템의 동작 중 LRISE의 설정 정보가 변경되었을 경우 WRISE를 통한 LRISE 요청 작업은 변경된 설정 정보에 의해서 잘못된 정보를 전송하거나 서버

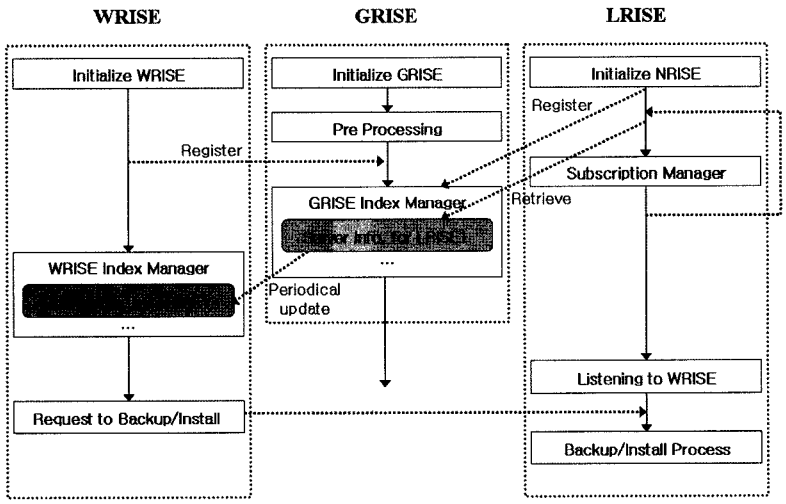

그림 6 자동 설정 변경 과정

의 오류를 일으킬 수 있다. 이러한 오류 방지 기능은 사용자나 관리자의 체계적인 관리가 어려운 분산 환경에서 매우 중요하다고 할 수 있다. RISE는 서버 간 자동으로 설정 정보를 동기화하는 기능을 제공하며, 자세히 살펴보면 다음과 같다(그림 6).

① GRISE는 각 서버의 정보를 수집하는 인덱스 서버의 역할을 수행하며 등록기능과 업데이트 기능을 통해 서버의 설정 변경을 저장한다.

② WRISE와 LRISE는 설정 파일 상의 GRISE 파라미터 셋팅 여부에 따라 GRISE에 등록을 한다.

③ 또한 등록 후에도 Subscription Manager를 통해 주기적으로 설정 파일의 변경을 확인하고 GRISE로 패치한다.

④ GRISE는 등록/변경되는 각 서버의 정보를 GRISE Index Manager를 통해 저장하고 주기적으로 WRISE 업데이트 한다.

4. 시스템 구현 및 성능 평가

4.1 구현 환경

WRISE는 웹 기반의 원격 백업/설치 등 복잡한 상호작용을 요구하기 때문에 JSP[6]와 Servlet[7]으로 구현하였고, Tomcat[8]을 웹 서버로 활용하였다. GRISE, LRISE, NRISE의 구현은 구현 편리성, 플랫폼 독립성, 가독성을 고려해 Python[9]을 사용하였고, PostgreSQL[10]를 데이타베이스 시스템으로 활용하였다. 이러한 RISE 시스템의 구현 환경을 표 2에 나타내었다.

RISE시스템에서는 PXE[12]와 램 디스크 기반 루트 파일 시스템을 이용해 아직 운영체제조차 설치되지 않은 초기상태의 노드의 MAC 주소를 수집하고, 현재 동

표 2 구현 환경

	WRISE	LRSIE, GRISE, NRISE
OS	RHEL AS 3.0 Update 4	
DBMS	PostgreSQL 8.0.3	
Program Language	JSP, Servlet, Javascript	Python2.2, pyXML[11]
Web Server	Tomcat 5.0	

작하고 있는 클러스터 내 노드의 백업 및 설치를 수행한다. 램 디스크 기반 루트 파일 시스템은 PXE 부팅 시 TFTP[13] 서버를 통해 대상 노드로 전송되고 커널은 이것을 마운트한다. 즉 파일 시스템을 물리적 메모리에 적재함으로써 로컬상의 저장 장치에 상관없이 시스템 동작을 가능하게 한다. RISE 시스템 상에서의 루트 파일 시스템의 구성은 다음과 같다.

- **네트워크 인터페이스 카드 드라이버 모듈**: RISE의 루트 파일 시스템은 현재 인텔, 리얼텍 계열의 랜카드를 인식하도록 구성되어 있다.
- **저장 장치별 디바이스 드라이버 모듈**: 클러스터의 저장장치 구성은 일반적으로 널리 활용되는 E-IDE, S-ATA, SCSI 방식의 저장장치를 사용한다.
- **이미지 백업/설치 도구 및 관련 라이브러리**: RISE는 현재 이미지 백업/설치 도구 중 파티션 기반 이미징 도구인 오픈소스 Partimage[14]와 dd[15] 지원한다. 이를 위해 partimage 바이너리 파일과 관련 라이브러리(압축, 인증), dd 명령어와 저장 공간의 효율을 위한 압축 명령을 포함한다.
- **Python 엔진 및 표준 라이브러리**: RISE는 Python 2.2로 구현되었으며 위의 실행을 위해 python 엔진

및 표준 라이브러리 파일이 포함된다.

4.2 성능평가

운영체제는 Redhat Enterprise Linux 3.0 update 4 이며 64비트 플랫폼을 기반으로 성능을 측정하였다. 그림 7은 실험 환경 구성도이다. 그림 7에서 알 수 있듯이, 사설망과 공인망이 혼합된 환경을 구현하기 위해서 2개의 LRISE 서버(LRISE1, LRISE2)가 GRISE 서버에 연결되어 있으며, LRISE1 서버에는 64개 노드로 구성된 클러스터 시스템이 48 포트 Gigabit 스위치 2개를 이용하여 연결되어 있다. LRISE2 서버에는 2개 노드가 연결되어 있으며, LRISE1서버와 LRISE2 서버는 별도의 사설망을 구성하여 각 클러스터 노드들과 연결되어 있다. 한편, 두 개의 LRISE 서버와 하나의 GRISE서버를 100Mbit 네트워크로 구성하였으며, LRSIE1서버에는 WRISE를 설치하여 공인망에 외부 공인망에 연결하였다. 각각의 LRISE서버는 클러스터를 구성하고 각자의 NRISE의 이미지 정보를 백업하여 가지고 있는 환경을 구성하였다. 이런 실험 환경을 통해 3-tier 구조가 가질 수 있는 장점, 즉 공인망과 사설망이 혼재되어 환경을 효과적으로 지원할 수 있는 것을 확인하였다.

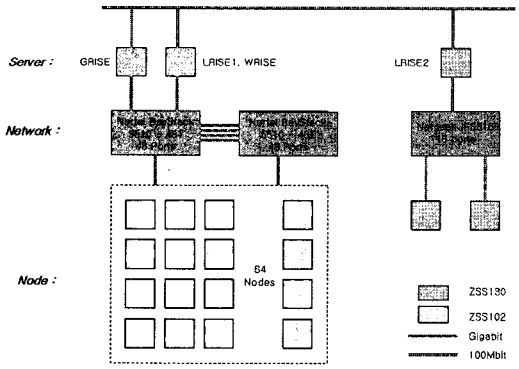

그림 7 실험 환경 구성도

RISE의 이미지 기반 원격 설치 기능의 성능 테스트를 위해서는 모두 64대의 삼성 스마트 서버 ZSS102 서버가 활용되었으며, 각 서버 노드의 사양은 표 3과 같다.

원격/백업 시간의 측정은 사용자 인터페이스(WRISE) 상에서 요청부터 완료시점까지의 시간을 초단위로 측정

표 3 NRISE 서버의 사양

Model	삼성 스마트 서버 ZSS102
CPU	Intel Pentium4 3.8GHz (EM64T)
RAM	3GB
HDD	S-ATA 250GB
LAN	Marvell Tech. Giagabit

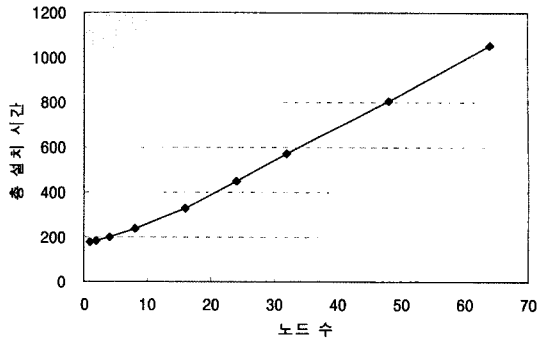

그림 8 RISE 이미지 총 설치 시간

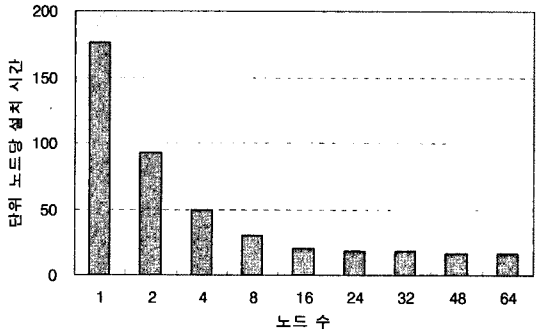

그림 9 단위 노드 당 설치 시간

하였으며, 각 노드에 설치할 이미지 파일을 확보하기 위해서 1대의 노드에 OS 및 응용 프로그램들을 설치하였고, 그 크기는 1.86GB 이다. 이미지 백업 및 이미지 설치 시에는, RISE에서 지원하는 2가지 방법 중 partimage 방법을 사용하여 실험하였다.

그림 8은 LRISE의 백업 이미지를 가지고 NRISE에 설치하는 총 시간을 측정한 그래프이다. 설치 노드 수는 1개, 2개, 4개 … 64개까지 늘려 가면서 측정하였고, 총 5회 테스트로 평균 값을 가지고 나타내었다. 총 64개의 노드에 모든 이미지 설치를 완료하는 데 평균 소요 시간은 17분 38초로 나타났다. 그림 8에서 알 수 있듯이, 노드 수가 증가하여도 64개의 노드까지 총 설치 시간이 선형적으로 증가함을 알 수 있다. 그림 9는 단위 노드당 설치 시간을 나타낸 그래프이다. 단위 노드당 설치 시간은 총 노드의 설치 시간을 설치 노드 수로 나누어 나타낸 값이다. 노드 수를 늘려가면서 동시 설치를 진행하였을 때 단위 노드당 설치 시간은 170초에서 16초까지 줄어드는 것을 볼 수 있었다. 또한 16노드 이후에는 단위 노드당 설치 시간이 16초 정도로 거의 일정하게 나타났다. 따라서 그림 8과 그림 9의 실험 결과를 통해, RISE의 이미지 기반 설치 성능이 설치 노드 수에 대한 확장성(Scalability)을 확보했음을 확인할 수 있다.

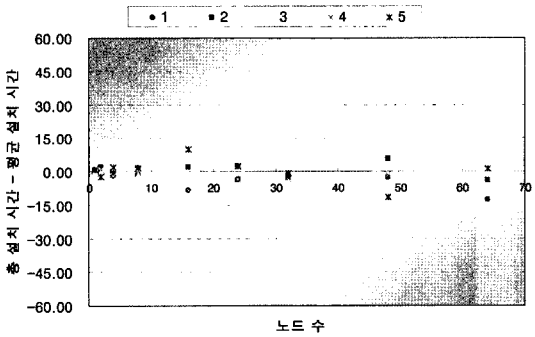

그림 10 설치 시간의 편차 그래프

그림 10은 5번의 총 설치 시간과 평균 설치 시간과의 편차를 나타내는 그래프이다. 이 편차를 통해서 RISE 시스템의 안정성을 알 수 있다. 실제 총 설치 시간과 평균 설치시간과의 차이가 적으면 적을수록 이 시스템이 안정적으로 동작하는 것을 알 수 있다. 64 노드까지 노드 수를 증가하며 측정한 결과 노드 수가 증가하더라도 편차가 12초 이하인 것으로 나타나 안정성을 갖는 것으로 나타났다.

마지막으로 LRISE에 오류를 발생시켜 GRISE를 통해 LRISE가 복구하는 과정을 테스트 하였다. 장애 발생 감지 시 LRISE의 ID를 통해 GRISE에 복구를 요청하고 GRISE는 해당 LRISE의 이미지 메타 데이타와 이미지 파일을 복구한다. LRISE의 복구 과정은 LRISE에 백업 시켜놓은 이미지의 양과 변경 시킨 셋팅들의 양에 따라 복구 시간은 차이를 보였지만 성공적으로 복구를 완료하였다. 이를 통해 RISE 시스템의 분산 환경에서 관리 노드의 안정성을 증명하였다.

5. 결 론

본 논문에서는 지리적으로 분산되어 있는 클러스터 시스템자원들을 효율적으로 활용하기 위한 3-tier 구조의 RISE 시스템을 제안하고 성능을 평가하였다. 실험결과 공인망과 사설망이 혼재하는 상황하에서 3-tier 구조는 효과적으로 분산 클러스터 시스템을 지원하는 것을 확인하였고, 자동설정 변경 동기화 기능을 통하여, 분산 환경에서의 RISE 시스템의 안정성을 확보했음을 확인하였다. 64 노드를 이용한 RISE의 성능실험에서, 1.86GByte의 시스템 이미지를 확보하는데 5분 53초의 시간이 소요되었고, 이를 64 노드에 설치하는 데 평균 17분 38초가 소요되었다. 초기 한 개 노드의 OS 설치시간이 25분 소요되었고, RISE 시스템 설치 및 구동에 5분이 소요된 것을 고려하면, 64개 노드 클러스터 시스템을 병렬처리 서비스가 가능하도록 하는데 소요되는 총 시간은 58분 31초 정도임을 알 수 있다. 만약, 1,024 노드의 클러스터 시스템을 설치한다고 할 때는 이미지를 활용한 원격 설치 단계를 64노드씩 16번을 반복설치 한다고 가정할 수 있다. 따라서, 1,024 노드를 역시 병렬처리 서비스가 가능하도록 준비하는데 소요되는 시간은 19,081초 (25분 + 5분 + 5분 53초 + 17분 38초 x 16), 즉 5시간 20분 정도임을 알 수 있다.

이러한 실험 결과들을 통하여 RISE 시스템은 점점 대형화되고, 복잡한 네트워크 망을 통해 구성되는 클러스터 시스템의 초기 설치 및 장애 복구 등에 효율적으로 활용될 수 있음을 확인할 수 있었다. 추후 연구를 통하여, 보다 다양한 OS에 대한 지원과 분산 환경에 적합한 보안 기능을 강화해 갈 것이다.

참 고 문 헌

[1] Top500 supercomputer sites, http://www.top500.org/
[2] Martin Hamilton. Red Hat Linux Kick-Start HOWTO, http://wwwcache.ja.net/dev/kickstart/Kick-Start-HOWTO.html
[3] J. Squyres, S. Scott, M. Chase-Salerno, S. Dague, N. Gorsuch (Open Cluster Group), Open Source Cluster Application Resources (OSCAR). http://oscar.sourceforge.net.
[4] M.J. Katz, P.M. Papadopoulos, G. Bruno, Leveraging standard core technologies to programmatically build Linux cluster appliances, Fourth IEEE International Conference on Cluster Computing, Chicago, IL, September 2002, pp. 47-53. appliances, Fourth IEEE International Conference on Cluster Computing, Chicago, IL, September 2002, pp. 47-53.
[5] B. Finley, S. Dague, M. Chase-Salerno, D. Frazier, System Installation Suite (SIS). http://sisuite.org.
[6] JavaServer Pages, http://java.sun.com/products/jsp/
[7] Servlet, http://java.sun.com/products/servlet/
[8] Apache Software Foundation, http://jakarta.apache.org/tomcat
[9] The Python Language Home Page, http://www.python.org
[10] PostgreSQL Home Page, http://www.postgresql.org
[11] XML package for Python, http://pyxml.sourceforge.net/
[12] Intel Corporation, Preboot execution environment (pxe) specification, http://www.intel.com/design/archives/wfm/downloads/pxespec.htm
[13] Sollings K. R., Trivial File Transfer (TFTP) Protocol, Version 2, Internet Request for Comments (RFC) July 1992.
[14] Partimage, http://www.partimage.org/
[15] Sam Chessman, http://www.linuxjournal.com/article/1320

박 두 식

1985년 2월 연세대학교 공과대학 전자공학과 졸업(학사). 1996년 2월 한국과학기술원 정보통신공학과 졸업(석사). 1986년 12월~현재, 삼성전자주식회사 근무. 관심분야는 HPC system, Cluster, LINUX customization

양 우 진

1996년 2월 성균관대학교 물리학과 졸업(학사). 1998년 2월 성균관대학교 물리학과 졸업(석사). 1998년 2월~현재, 삼성종합기술원 근무. 관심분야는 병렬처리 및 전산물리학

반 민 호

2001년 2월 서울시립대학교 전자공학부 졸업(학사). 2001년 3월~현재, 삼성전자주식회사 근무. 관심분야는 병렬컴퓨팅과 OS 가상화

정 갑 주

1984년 2월 서울대학교, 컴퓨터공학과(학사). 1986년 2월 서울대학교, 컴퓨터공학과 인공지능(석사). 1996년 2월 New York University, Computer Science 박사. 1995년 12월~1997년 8월 University of Florida, Post Doc. 1997년 8월~2001년 건국 대학교, 컴퓨터 공학과 조교수. 2001년~현재 건국 대학교, 인터넷&멀티미디어 공학부 부교수. 2006년~현재 BK21 u-Science 기반 신기술융합 사업단 단장 관심분야는 Grid Computing, e-Science, Data Integration, 및 분산컴퓨팅

이 종 현

2003년 2월 건국대학교 컴퓨터 공학과 학사. 2005년 2월 건국대학교 컴퓨터 공학과 석사. 2005년 3월~현재 건국대학교 컴퓨터 공학과 박사과정. 관심분야는 Grid Computing, e-Science, Linux Programming

이 상 문

1985년 12월 William Penn College, Computer Science & Mathematics(학사). 1989년 5월 Iowa State University, Mathematics(석사). 1995년 12월 State University of New York at Stony Brook, Applied Mathematics(박사) 1995년 12월~현재, 삼성종합기술원 근무. 관심분야는 Grid Computing

이 창 성

1996년 2월 서울대학교 공과대학 항공우주공학과 졸업(학사). 1998년 2월 서울대학원 공과대학 항공우주공학과 졸업(석사). 2004년 2월 서울대학원 공과대학 항공우주공학과 졸업(박사). 2004년 2월~현재, 삼성전자주식회사 근무. 관심분야는 GRID Computing, 병렬처리 알고리듬, Cluster Management

신 순 철

1999년 2월 수원대학교 전산학과 졸업(학사). 2001년 2월 수원대학교 전산학과 졸업(석사). 2001년 2월~현재, 삼성종합기술원 근무. 관심분야는 High Performance Computng

이 인 호

1982년 2월 경북대학교 공과대학 전자공학과 졸업(학사). 1984년 2월 경북대학원 공과대학 전자공학과 졸업(석사). 1983년 10월~현재, 삼성전자주식회사 근무. 현재 서버팀장. 관심분야는 HPC system, Home storage system