논문 2016-11-30

신뢰성 있는 멀티스택 기반의 가상화된 데이터 동시공유 시스템의 구현

(An implementation of reliable data sharing multi-stack system in virtualized environment)

한규종, 전동운, 김두현*
(Kyujong Han, Dongwoon Jeon, Doohyun Kim)

Abstract : In this paper, we present an architecture for the fault isolation by applying virtualization-based multi-stack technologies. We propose the simultaneous sharing and switching mechanism using virtualied serial communications. Each guest OS has its own virtual serial device. The distribution module provides communications between the guest OS's through the virtual serial devices and simultaneously detect the liveness of the guest OS. The suggested mechanism has been implemented in VirtualBox and shows satisfactory performance in transmission speed and data sharing capability with virtual RS232.

Keywords : Virtualization, Multi-stack, Fault tolerance, VirtualBox, Serial I/O, RTOS, eCos

Ⅰ. 서 론

최근 클라우드와 사물인터넷 (IoT: Internet of Things) 기술의 발달로 다양한 단말에서 발생하는 대량의 데이터가 네크워크를 통해 전송된다. 데이터의 생성에서부터 저장되는 과정을 살펴볼 때 저장된 이후 데이터는 안정적이지만 데이터의 생성 및 수집 단계에서의 데이터 안정성은 상대적으로 취약하다. 따라서 데이터를 수집 및 Reaction을 담당하는 에지 (edge)단에서의 시스템의 신뢰성을 높이기 위한 방안이 필요하다. 에지 단은 클라우드 단말 디바이스와 클라우드 서버 사이에 위치하는 게이트웨이의 형태로 데이터를 수집 및 분석하는 기능을 수행한다. 이러한 에지 단의 하드웨어는 일반적으로 저전력이면서 소형의 임베디드 시스템의 특성을 갖는데 본 논문에서는 멀티스택 기반의 가상화 기술을 응용하여 시스템의 신뢰성, 가용성 및 안정성을 높이는 방안을 제안한다.

가상화 기술은 통합된 스토리지와 컴퓨팅 하드웨어, 네트워크 등을 필요한 만큼 활당하여 논리적인 서버를 구성해 제공함으로써 인하여 효율성 및 유연성과 경량화, 비용 절감이라는 장점을 가진 기술이다 [1]. 또한, 현재의 가상화 환경은 주로 클라우드 환경과 서버 운용에 있어 중요한 기술 중 하나라고 할 수 있다. 단일의 물리적인 장치 내에 다수의 가상화된 스택 (stack)을 운용함으로서 하드웨어의 효율성과 단가 절감의 효과가 있다. 본 논문은 이러한 클라우드 환경과 서버 운용에서 사용되어지는 가상화된 운영체제에 고장 감내 기법을 적용하여, 임베디드 시스템의 중단 없는 운용방법을 실현하는 것과 관계가 깊다. 즉, 이러한 고장 감내 시스템이의 실현을 위해서는 가상화된 게스트 (Guest) 운영체제 간 통신과 호스트 운영체제와의 다대일의 동시공유를 필요로 하는데, 본 논문에서는 호스트와 게스트 간 데이터 동시공유와 가상화된 운영체제 사이의 통신을 위한 모듈을 구현하여 고장 감내 시스템에 필요한 요구사항을 연구한다. 또한 구현의 결과로 가상화된 시스템 간 태스크 (task) 스위칭 성능을 측정하여 고장 감내 시스템의 운용성을 평가하였다.

*Corresponding Author (doohyun@konkuk.ac.kr)
Received: 9 May 2016, Revised: 13 June 2016, Accepted: 1 Aug. 2016.
K.J, Han, D.W, Jeon, D.H, Kim: Konkuk University
※ 본 연구는 민군겸용기술사업 (Dual Use Technology Program) 지원을 받아 수행되었음 (UM13018RD1).

ISSN : 1975-5066
http://dx.doi.org/10.14372/IEMEK.2016.11.5.259

표 1. 가상머신별 취약점 수 분석
Table 1. Analysis of security of vulnerability

OS / Analysis	VMware	Xen	VirtualBox
Security of vulnerability	91	23	3

Ⅱ. 관련 연구

1. 가상화 기술과 멀티스택

가상화 기술은 호스트 (host)로 부르는 하나의 물리적인 시스템을 토대로 단일 혹은 다중의 가상 시스템 환경을 구성하여, 각각의 가상화 환경에서 발생될 수 있는 소프트웨어적 오류에 대해서 상호 독립성을 유지하면서 수행이 가능한 환경을 의미한다 [2]. 본 논문에서는 가상화를 지원하기 위한 가상머신으로 현재 상용되는 VMware, VirtualBox, Xen 이렇게 세 종류의 가상머신을 CVE (Common Vulnerability and Exposures) 기반으로 취약점을 분석하여 이 중 보안이라는 특성이 가장 강인한 것으로 보고되어 있는 VirtualBox를 기반으로 시스템을 구현하였다 [3]. 표 1은 현재 가장 많이 활용되는 가상머신별 취약점을 수치화한 것으로 공개적으로 알려진 정보보안의 취약점 혹은 위험 노출에 대한 정보를 나타낸다.

가상화의 핵심 기능 중 VMM (Virtual Machine Monitor) 또는 하이퍼바이저 (Hypervisor)라고 불리는 소프트웨어 계층은 다중 가상화 환경을 제공하며, 생성된 VMM에서 실행되는 가상머신을 스택 (stack) 혹은 게스트 도메인 (Guest Domain)라고 부른다.

가상화의 종류는 크게 두 가지로 구분할 수 있는데, 첫 번째로 게스트 운영체제를 수정하지 않고 기존 소프트웨어를 그대로 수행할 수 있지만 CPU, I/O 인터페이스 등의 에뮬레이션 오버헤드가 발생되는 전가상화 (Full-Virtualization)와 두 번째로, 오버헤드를 줄이기 위해 에뮬레이션을 하지 않고 게스트 운영체제와 디바이스 드라이버를 VMM이 직접 제공해주는 반가상화 (Paravirtualization)로 구분 할 수 있다. 본 논문의 구현에서 사용된 VirtualBox는 오라클에서 개발 및 배포되는 가상화 도구로 Guest OS의 응용프로그램과 디바이스 드라이버의 수정을 요구하지 않는 전가상화를 제공한다 [4].

2. 네트워크 I/O 구조

VirtualBox의 VMM은 다양한 네트워크 I/O 지원

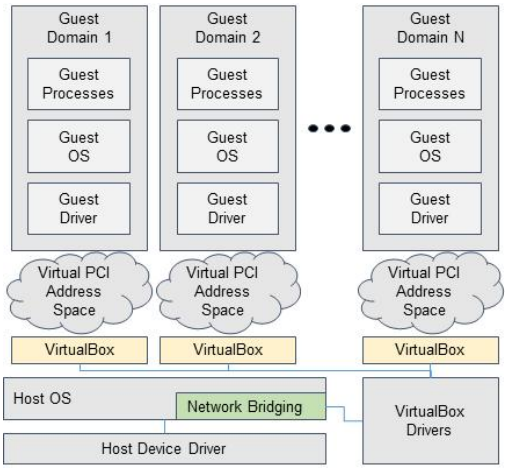

그림 1. VirtualBox의 네트워크 I/O 구조
Fig. 1 Network I/O structure on VirtualBox

방식을 제공하고 있다. 그림 1은 VirtualBox에서 지원하는 네트워크 I/O 구조로 VMM은 물리 PCI 디바이스 제어 환경을 제공하기 위해 게스트 도메인에 가상의 PCI 주소 공간을 제공한다. 제공된 PCI 주소 공간은 물리 PCI 디바이스 제어와 같은 기능을 갖게 된다. 그림 1은 VirtualBox의 네트워크 I/O 구조를 나타낸 것으로, 네트워크 데이터의 송수신 경로는 게스트 디바이스 드라이버의 송신 요청을 VMM이 인지하면 VirtualBox 드라이버를 통해 호스트 운영체제 내의 네트워크 브리지에 전달하고 이후 호스트 장치에서 직렬통신을 수행하게 된다.

3. RTOS와 eCos

RTOS (Real-time OS)는 할당된 작업을 주어진 시간 내에 수행하는 것을 목적으로 하는 운영체제로 본 논문에서는 예측이 가능한 시스템 함수를 사용하여 유저가 원하는 데이터 출력을 제공하는 것으로 의미를 좁혀서 적용하였다. RTOS에서 제공되는 다중 태스크 (Multi Task) 구조를 이용해 시간 제어, 태스크들 간의 통신, 동기화, 스케줄링 (Scheduling) 등의 매커니즘 (mechanism)을 제공하는 실시간 시스템을 구현한다 [5].

본 논문에서는 리눅스 시스템에서 가상머신을 이용한 가상화된 RTOS를 탑재하여 실시간성 부여하고 예상치 못한 변수 제어, 정확한 출력 요구 조건 설정, 출력 제어를 할 수 있도록 보장하였으며, 커널의 수정 없는 고장감내를 위한 시스템의 구현

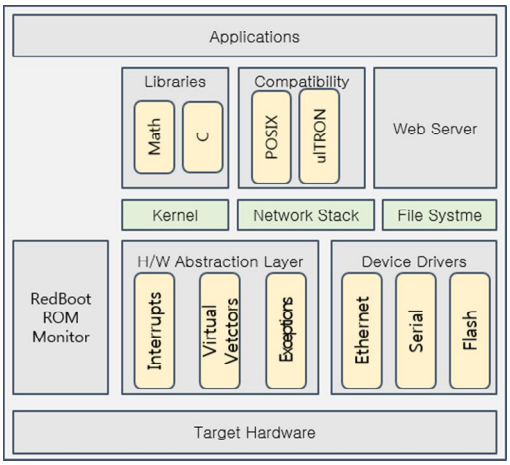

그림 2. 호스트 시스템에 탑재된 eCos 구조
Fig. 2 eCos Structure on Host system

성능 측정 방법을 제안하고 이를 검증한다. 본 논문에서는 실시간 멀티 스택 가상화를 위해 RTOS의 한 종류인 eCos를 게스트 도메인에 포팅하여 시스템을 구현하였다. eCos는 경량의 오픈 소스 실시간 운영체제로 신뢰성 있는 임베디드 소프트웨어 개발을 위해 활용된다. 현재 상용화된 여타의 RTOS들과 유사한 실시간성을 보장하며, GNU 툴과 Cygwin 패키지를 사용하여 개발되었다. eCos는 적은 시스템 자원 (Resource)의 사용, HAL (Hardware Abstraction Layer) 등과 같은 설정 기능을 지원한다. 그림 2는 eCos를 활용하여 구성한 임베디드 리눅스 시스템의 아키텍처의 구조를 나타낸다.

4. 고장 감내 시스템

고장 감내 시스템은 치명적인 결과나 막대한 비용이 발생하는 임무시스템 (Mission-critical System) 또는, 메인 프레임급 금융 시스템 서버와 대기업 관리 서버 등의 고가용 시스템에 고장이 발생하는 경우 빠르게 대처하기 위한 기술이다 [6].

고신뢰성의 시스템을 위한 고장 감내 시스템은 하드웨어 고장 감내 시스템과 소프트웨어 고장 감내 시스템으로 나눌 수 있다. 하드웨어적인 고장의 감내 방법은 작업을 수행하는 시스템과 동일한 중복된 (redundant) 하드웨어를 이용해 부분적 혹은 전체적으로 정상적인 기능을 수행하도록 전환하는 것을 의미한다. 소프트웨어 기반의 고장 감내 시스템은 부가적인 하드웨어를 필요로 하지 않으며, 시스템 내에서 발생하는 고장에 대하여 소프트웨어적으로 감지 및 대처한다.

다양한 분야에서 시스템의 고장에 대비하기 위한 고장감내를 위한 연구가 진행 중이며, 크게 운영체제 영역의 수정을 이용한 방식, 롤백 (Roll-Back) 방식, 이중화 방식, 복제 방식 및 다양화 방식 등과 같은 다양한 형태의 복구 기법이 연구되고 있다 [7].

본 논문에서는 하나의 물리적 시스템에 다수의 가상머신을 설치하고 가상 머신 간 통신환경 구축을 통한 고장의 감지 및 복구 기법을 제안한다. 이러한 고장 감내 기법은 소프트웨어의 재사용성을 높이고 가상화를 이용한 설치 및 수정의 용이함을 제공하고 유휴자원을 활용한 시스템의 가용성을 높이면서 신뢰성 높은 시스템을 구축하는 이점을 갖는다.

Ⅲ. 테스트 환경 구성과 시스템 구현

1. 환경 구성

동시 공유를 위한 가상머신을 구현하기 위하여 본 논문에서는 호스트 운영체제인 우분투에서 가상화된 eCos를 사용하고자 아래와 같은 운영체제 및 소프트웨어적인 개발 환경에서 가상머신을 구현하였다.

- Ubuntu 11.04 LTS
- VirtualBox 4.2.6 r82870 (Open Source)
- eCos release 3.0 (RTOS)

IoT 게이트웨이를 가정한 작은 크기의 컴퓨팅 머신으로 저전력을 소모하면서 가상화에 필요한 컴퓨팅 자원을 보유한 하드웨어로 본 논문에서는 Intel사의 NUC 모델 (Intel core i5 4250U 프로세서, 4GB Memory)을 사용하였다.

2. 호스트 내에 모듈 구현

고장 감내 기법을 구현하기 위해서 가상화된 운영체제간의 통신을 위한 모듈과 동시공유를 위한 모듈의 구현이 필요하다. 먼저 동시공유를 위한 분배모듈의 구현 방법은 호스트 내에 구성하는 방법과 가상머신에 탑재하는 두 가지 방법이 있다. 본 논문에서는 기존의 연구 방법을 참조하여 가상화된 운영체제 간 통신성능이 우수하며 적은 시스템 부하를 유발하는 호스트 내부에 모듈을 구현하는 방법으로 시스템을 구성하였다 [6].

가상머신에 VirtualBox를 호스트 운영체제로 설치하고 VirtualBox내에 분배모듈과 통신을 위한 가상

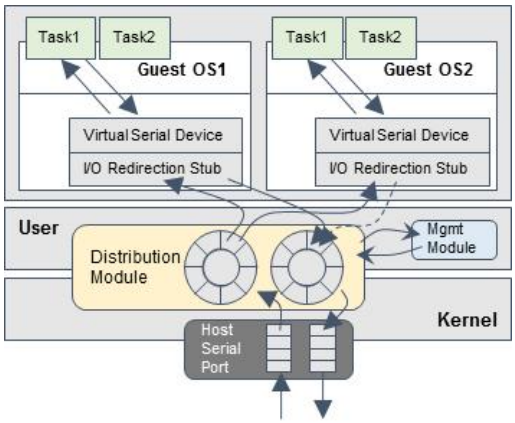

그림 3. 호스트 운영체제 내의 분배모듈 구조
Fig. 3 Distribution module structure in Host OS

시리얼 디바이스를 구현하였으며, 이는 가상머신 내에 게스트 운영체제인 eCos가 호스트와의 가상 통신을 통하여 게스트 운영체제 간 데이터 공유와 외부 장치와의 데이터 통신을 할 수 있도록 한다.

호스트 내에 모듈로 구현된 분배모듈은 두 가상화된 게스트 운영체제들과 동시에 공유된다. 유저영역에 구현된 분배모듈은 VirtualBox 내부의 가상 시리얼 디바이스와 가상 직렬 통신을 하도록 구현하였으며, 분배모듈에서는 각각의 게스트 운영체제의 태스크 구분을 위한 UUID (VM-UUID)와 TASK ID를 참조하여 두 게스트 운영체제를 구분한다. 분배모듈에서는 외부장치에서 직렬포트를 통해 들어오는 데이터를 복제하여 각각의 게스트 운영체제의 I/O Redirection Stub에 전달하는데, TASK ID, UUID (VM-UUID), 플래그 및 타임스탬프(time stamp)에 포함된 데이터를 이용한다. 그림 4는 호스트 내부에 구현된 분배모듈과 게스트 운영체제간의 동시공유 구조를 나타낸다. 외부 장치에서 오는 데이터는 호스트의 시리얼 포트를 통해 수신되며, 수신된 데이터와 가상머신에 설치된 eCos에서 출력되는 데이터는 유저영역에 구현된 분배모듈로 전송된다. 분배모듈의 역할은 외부장치에서 전송되는 데이터를 복제하여 각각의 게스트로 전달하고, 게스트는 전송된 데이터를 처리한 이후 결과 값을 다시 분배모듈로 리턴한다. 리턴된 데이터는 분배모듈에서 게스트들의 상태체크 데이터와 함께 처리하여 두 개의 게스트 중 마스터에 해당하는 게스트의 데이터를 외부장치로 송출하는 역할을 수행한다.

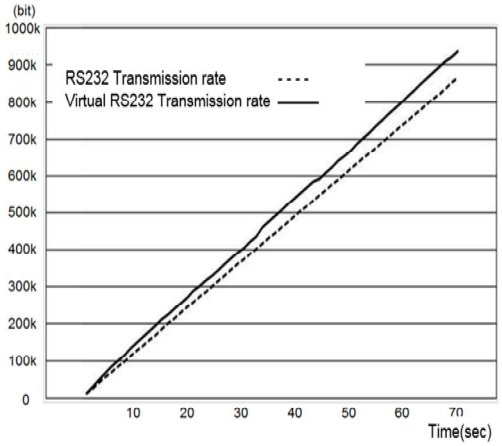

그림 4. RS232 시리얼과 가상 직렬 시간에 따른 누적 전송량 비교
Fig. 4 Accumulated transmission speed comparison of physical RS232 and virtual RS232

3. 가상머신 내에 가상 직렬 장치 구현

앞 절에서 서술한 모듈 중 분배모듈은 가상머신 내에 구현된 가상 직렬 장치와 통신한다. 그림 3에서 분배모듈은 동시공유 뿐 아니라 Mgmt Module을 통해 게스트 운영체제 간의 통신을 지원한다. 본 논문에서는 VirtualBox내에 가상 네트워크 장치를 구현하여 가상환경에 맞는 고장 감내 통신 환경을 구축하였다. 고장감내에 필요한 게스트 운영체제 간의 통신은 1024바이트 이하 정도의 전송을 빈번하게 사용하는 통신 특성에 맞게 지연시간을 최소화하여 전달한다.

가상의 직렬 장치를 통해 구현된 고장감내 방법은 Mgmt Module에서 서브로 운용되는 게스트 운영체제 내에서 고장을 감지하여 특정 데이터 값을 송출하였을 때 즉시 메인 게스트 운영체제의 데이터 대신 서브 게스트 운영체제의 데이터 값으로 스위칭하여 외부장치로 송출 하도록 구현하였다.

Ⅳ. 실험결과 및 분석

1. 동시공유를 위한 가상 직렬 장치 전송속도

그림 4는 가상머신 내에 구현된 가상 직렬 장치와 기존 RS232 시리얼 장치와 전송속도 비교 그림이다. 실선은 가상머신과 호스트 간 직렬 통신의 시간에 따른 누적 전송량을 나타내며, 점선은 RS232 시리얼 장치의 속도를 115200bps로 설정했을 때를

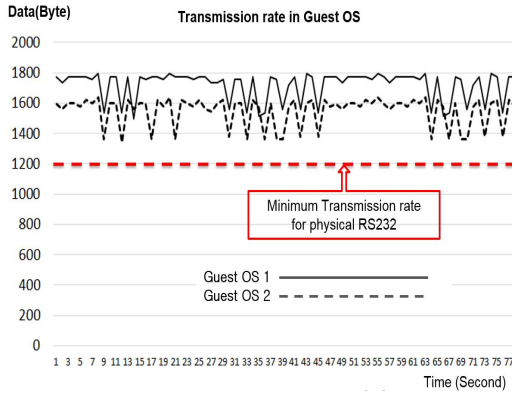

그림 5. 동시공유 통신 성능
Fig. 5 Transmission performance in data sharing

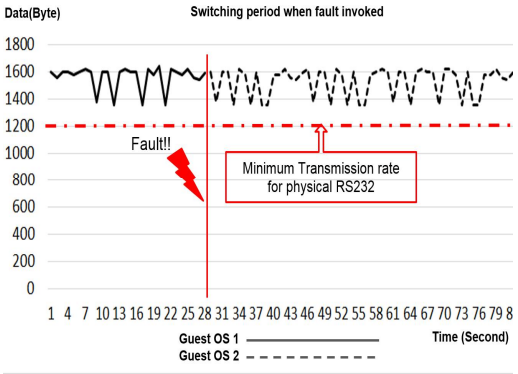

그림 6. 고장 발생 시 스위칭 성능 평가
Fig. 6 Evaluation of switching when fault invoked

기준으로 나타낸 시간에 따른 누적 전송량을 나타내고 있다. 각각의 시간에 따른 누적 전송량을 보았을 때 가상 시리얼 전송 속도가 RS232 시리얼의 전송 속도보다 높은 성능을 나타내는데 가상 시리얼의 동작방식은 RS232와 동일하게 동작하도록 구현하였으나 전용 네트워크 포트를 이용하기 때문에 가상머신과 호스트간의 지연이 발생하지 않아 실제 측정에서는 물리 장치보다 높게 측정되었다. 이러한 결과는 가상으로 구현된 게스트 간 직렬 장치의 전송속도가 물리적인 시리얼 장치의 전송속도의 요구조건을 만족하면서 실시간 데이터 송수신이 가능한 것을 의미한다.

2. 동시공유 통신 성능

그림 5는 각각의 게스트 운영체제와 호스트간의 동시공유 통신성능을 나타낸 것이다. 전송량은 각각의 게스트 운영체제에 상이하게 구현하였으며, 외부장치와의 통신을 위한 최저 데이터 전송량을 만족하는 성능을 나타내는데, 최저 데이터 전송량은 1024바이트이나 고장 감내 시스템의 실시간성 보장을 위해 최소 데이터 전송량을 1200바이트로 설정하였다. 데이터 전송량이 그 이하로 떨어질 경우 고장 감내 시스템의 실시간성 속성과 전송 성능을 만족하지 못하게 된다. 다시 말해, 구현된 시스템의 내부 데이터 전송속도가 일정 수준에 도달하지 못하는 경우 고장감내를 위한 최소한의 성능을 갖추지 못했음을 의미한다. 본 논문에서는 게스트 운영체제의 스위칭을 통해 하나의 게스트 운영체제의 데이터 값을 외부장치로 전달하도록 구현하였다.

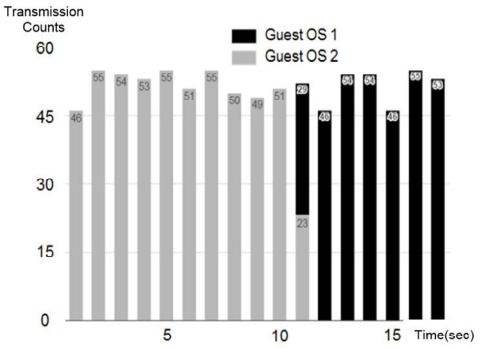

그림 7. 외부장치에서 패킷을 수신한 횟수
Fig. 7 Received packet counts from external network

3. Fault 발생과 스위칭 성능 평가

고장 감내 시스템에서 Fault가 감지되었을 때 호스트 내에 모듈에서 스위칭이 발생한다. 마스터 혹은 액티브에 해당하는 가상화된 운영체제에서 호스트로 보내지는 데이터는 모듈에 의해 slave 혹은 스탠바이의 가상화된 운영체제의 데이터로 스위칭되어 데이터를 전송한다. 그림 6은 고장 발생 시 메인으로 동작하던 게스트 운영체제에서 송신되던 데이터를 서브로 대기하던 게스트 운영체제의 데이터로 변경해서 전송하는 스위칭을 나타낸 그림이다. 게스트 운영체제 1에서 외부 장치로 전송하던 데이터를 고장이 발생한 이후 게스트 운영체제 2에서 발생한 데이터로 전환하여 외부장치로 전송되는 것을 확인 할 수 있다.

그림 7은 외부장치에서 매 초마다 수신한 데이터

횟수를 나타낸 그래프이며 고장 발생 순간에 각각의 게스트 운영체제의 데이터를 받아오는 것을 확인 할 수 있다. 매 초마다 약 50회 정도의 데이터를 수신하는데, 고장 발생 시 두 개의 게스트 운영체제에서 발생한 데이터를 모두 수신하였으며, 이때 고장이 발생한 것을 확인 할 수 있다. 각각의 게스트 운영체제의 전송 횟수는 초당 50회 이상임을 확인하였으며, 고장의 발생 순간에도 두 게스트 시스템의 데이터 전송 횟수를 합산하여 50회 이상으로 측정되어 고장이 발생하여 태스크 수행이 다른 노드로 전이되는 과정에서도 안정적으로 데이터를 송신하는 것을 확인할 수 있다.

V. 결 론

본 논문은 가상화 기반 멀티스택 기술을 적용하여 스택간의 오류 고립화와 소프트웨어의 재사용성을 높이기 위한 연구의 결과로 가상화 시스템에 복수의 가상머신을 설치하고 가상머신 간의 통신 속도 측정 및 고장 발생 시 중단 없는 태스크의 유효성을 가상머신에서의 전송 속도 및 초당 데이터 전송횟수를 측정하여 검증하였다. IoT 에지 단을 위한 가상머신 기반의 멀티스택 구조를 제안 및 구현하고 스택 간 가상화된 내부통신 (RS232) 채널을 구축하고 데이터의 전송속도를 측정하였다.

본 논문에서는 가상화된 게스트 운영체제들과 호스트 운영체제 간의 동시공유 전송을 위한 네트워크 I/O 구조를 지원하는 가상직렬장치와 분배모듈을 구현하였다. 그리고 네트워크 I/O를 동시에 공유하도록 가상머신 내의 가상장치의 구성을 살펴보았으며, 입출력 Redirection Stub과 분배모듈을 설계하고 구현하였다. RTOS인 eCos를 탑재하여 게스트 운영체제 내에서 실시간 처리를 지원하였고, 가상 직렬장치는 물리적 직렬장치 입출력의 성능을 만족하도록 구현하였다. 또한 가상화된 운영체제 간 고장감내를 위한 게스트 운영체제 사이의 통신과 스위칭의 설계 및 성능평가를 통해 가상화 환경에서 고장감내 시스템의 구현을 위해 고려되어야 할 통신 성능 및 모니터링 역할을 하는 모듈을 구현하여 스위칭이 원활하게 작동되는 것을 확인하였다. 이를 통해, 가상화된 환경에서의 고장감내 시스템의 정상적인 동작을 확인하였다.

본 연구를 통해 임베디드 환경에서 가상화 기법의 적용을 통한 고장감내 기능의 구현은 최근 활발히 연구되고 있는 클라우드 시스템에서의 에지 노드 혹은 스마트 게이트웨이 등의 가용성 및 신뢰성을 높이는 방안으로 활용할 수 있다. 또한, 하드웨어적인 고장 감내 시스템이 아닌 소프트웨어적인 고장감내 시스템의 구축을 통해 저비용과 경량화에 이점이 있으며 향후 하드웨어와 통신기술의 발달로 설치 및 적용이 용이한 고장감내 시스템의 구축에서 가상화 환경이 주효할 것으로 기대된다.

References

[1] M. Rosenblum, T. Garfinkel, “Virtual Machine Monitors: current Technology and Future Trends,” IEEE Computer, Vol. 38, No. 5, pp. 39-47, 2005.

[2] K. Adams, O. Agesen, “A comparison of software and hardware techniques for x86 virtualization,” Proceedings of the 12th International Conference on Architectural Support for Programming Languages and Operating Systems), Vol. 40, No. 5 pp. 2-13, 2006.

[3] J.Y. Kim, H.J. Kim, C.S. Park, M.J. Kim. “Analysis of weakness on cloud computing environment virualization technology,” Korea Institute Of Information Security And Cryptology, Vol. 19, No. 4, pp. 72-77, 2009 (in Korean).

[4] H.W. Jin, J.S. Kim “A structure for network I/O on virtualized platform,” The Journal of The Korean Institute of Communication Sciences, Vol. 29, No. 9, pp. 38-43, 2012 (in Korean).

[5] E.C. Shin, B.W. Choi, “Implementation of a Mobile Robot Control Platform using Real-Time Embedded Linux,” Journal of Institute of Control, Robotics and Systems, Vol. 12, No. 2, pp. 194-200, 2006 (in Korean).

[6] J.H. Yoo, K.J. Han, “The Design of Fault Tolerant PSTR Using Virtualization Techniques on the Embedded System,” KIPS Transactions on Computer and Communication Systems, Vol. 3, No. 12, pp. 443-448, 2014 (in Korean).

[7] S.Y. Jo, “Design and Implementation of Fail Recovery Process on Highly-Reliable Embedded Linux System,” Journal of Security Engineering, Vol. 11, No. 1, pp. 89-100, 2014 (in Korean).

Kyujong Han (한 규 종)

2013:, B.E. degree in Division of Information and Communication from Baekseok Univ, Korea.

2015: M.S. degree in Internet & Multimedia Engineering from Konkuk Univ, Korea.

Current: Engineer at Datastreams.

Research Interests: realtime system.

Email: karjensia@gmail.com

Dongwoon Jeon (전 동 윤)

2005: B.E. degree in Internet & Multimedia engineering from Konkuk Univ, Korea.

2008: M.S. degree in Internet & Multimedia engineering from Konkuk Univ, Korea.

2015: Ph.D. degree in the school of Computer, Information & Communications Engineering at Konkuk University, Korea.

Current: Researcher at NICEInfo.

Research Interests: embedded software.

Email: birdybuddy@gmail.com

Doohyun Kim (김 두 현)

1985: B.E. degree in Computer Science from Seoul National University, Korea.

1987: M.S. degree in Computer Science from KAIST, Korea.

2003: Ph.D. degree in Computer Science from KAIST.

Current: Professor, Department of Internet & Multimedia Engineering Konkuk University.

Research Interests: embedded software, Real-time OS.

Email: doohyun@konkuk.ac.kr