

# 능동 네트워크를 위한 Enode 운영체제 설계 및 구현

장승주\*, 나중찬\*\*, 이영석\*\*

Design and Implementation of the Enode Operating System for the Active Network

Jang, Seung Ju\*, Na, Jong Chan\*\*, Lee, Young Suk\*\*

---

본 논문은 2002년도 ETRI 능동보안기술연구팀의 연구 과제 지원에 의하여 이루어졌음  
이 논문은 2003년도 Brain Busan 21사업에 의하여 지원되었음

---

## 요약

본 논문은 능동 네트워크를 동작하게 해주는 핵심 모듈인 Enode 노드 운영체제 구조 및 세부 기능들의 내용을 정의한다. 본 논문에서 제안하는 Enode 노드 운영체제는 능동 네트워크 환경에 적합하도록 설계하였다. 또한 "실행 환경"(Execution Environment : EE)에서 편리하게 사용할 수 있도록 인터페이스를 설계하였다. 본 논문에서 제안하는 Enode 노드 운영체제는 도메인을 중심으로 핵심적인 기능 설계에 주력하였다. Enode 운영체제는 Linux 운영체제 상에서 설계되었다. 또한 본 논문에서 제안하는 Enode 노드 운영체제의 인터페이스에 대한 실험을 수행하였다.

## ABSTRACT

This paper suggests Enode Operating Systems that is core part of active network. It is composed of five parts: domain, channel, thread, memory and file module. The domain and channel module among five parts are the main function. The remaining parts that are the thread, memory, and file are the supporting module for the domain and channel. The domain module manages active network. It creates and deletes domain data structure. The channel module has an inchan, outchan, and cutchan. We also test the Enode Operating Systems to verify suggesting concept of node Operating System

## 1. 서론

능동 네트워크에 대한 연구는 DARPA를 중심으로 현재 여러 기관들이 자체적인 그룹을 결성하여 활발한 연구 활동을 하고 있다.

현재 주요 연구 기관 및 대학에서 개발된 액티브 네트워크 구조는 여러 가지가 있다. 그 중에서 ANTS는 MIT에서 연구 개발된 "실행 환경"과 프로그래밍 환경을 제공하고, CANE은 Georgia Tech에서 연구중인 동적인 네트워크 변경을 처리할 수 있도록 하는 특징을 갖는다 [1, 2].

일반적으로 능동 네트워크 노드의 구조는 "실행 환경"(EE : Execution Environment)과 노드 운영체제의 두 부분으로 나누어진다. Bowman 이나 CANE의 경우가 이런 형태의 능동 네트워크 구조를 가진다. Odyssey 의 경우는 가장 하부에 호스트 운영체제가 상주하고, 이 운영체제를 기반으로 active OS가 있다. 그리고 이 active OS를 바탕으로 하여 EE가 상위에 동작한다. 그리고 EE 상위에 능동 응용 프로토콜이 동작한다 [19].

노드 운영체제는 DARPA에서 정의한 능동 네트워크 노드 구조에서 하부 구조를 차지한다. 노

드 운영체제 구조는 능동 네트워크 구조에서 원초적인 기능을 수행한다. Bowman 노드 운영체제는 이런 DARPA의 기능 규격을 충실히 만족하고 있다 [19]. Bowman 노드 운영체제는 다음의 3가지 추상적인 개념으로 정의되어 있다. : 채널, a-flow, state store 등이다 [19].

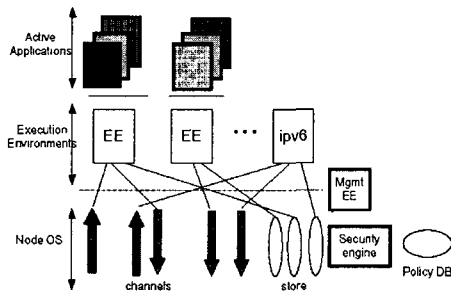


그림 1. 능동 네트워크에서 노드 운영체제 구조  
Fig. 1 Node Operating System Architecture in Active Network

노드 운영체제는 "실행 환경"과 물리적인 자원을 관리하는 실제적인 운영체제 사이에서 동작하는 모듈이다. 노드 운영체제는 복수의 수행 환경을 지원하고 모든 능동 노드에 공통적인 기능을 지원할 수 있어야 한다. 이러한 노드 운영체제에 요구되는 공통적인 기능은 각 EE가 패킷을 수신 및 송신하기 위한 채널 기능의 구현, 노드 자원들에 대한 사용권한을 결정하는 부분으로 구성된다 [3, 4, 5, 6].

본 논문에서 제안하는 노드 운영체제는 Enode(ETRI Node) 운영체제라고 부른다. 본 논문에서 제안하는 Enode 노드 운영체제는 능동 네트워크 노드에서 효과적인 자원 관리를 위한 목적으로 설계된다. 본 논문에서 제안하는 노드 운영체제의 기능은 도메인, 채널, 쓰레드, 메모리, 파일 모듈로 구성된다 [6, 7, 8, 9]. 이 중 본 논문에서는 도메인과 채널 기능을 중심으로 설계된 내용을 언급한다. 2장은 Enode 노드 운영체제 구조, 3장은 Enode 노드 운영체제 설계 내용, 4장 평가, 5장 결론의 순으로 언급한다.

## II. Enode 노드 운영체제 구조

Enode 노드 운영체제 구조는 그림 2와 같다. Enode 운영체제 상위에는 "실행 환경"이 존재한다. "실행 환경"은 능동 네트워크가 동작하는데 필요한 응용 프로그램(액티브 응용)과 여러가지 응용 프로그램이 수행하는데 필요한 사용자 인터페이스를 제공한다. Enode 노드 운영체제는 "실행 환경"에 대한 인터페이스를 가지고 있고, 이 인터페이스를 통해서 "실행 환경"과 Enode 노드 운영체제와 정보 전달을 할 수 있다. 일반적으로 "실행 환경"과의 인터페이스 역할을 수행하는 Enode 노드 운영체제 내의 모듈이 도메인이다. Enode 노드 운영체제에서 도메인을 통해서 패킷을 전달하기 위해서 채널 모듈과 상호 동작이 이루어진다. 그 밖에 Enode 노드 운영체제가 수행하는 역할은 대역폭, 메모리 등의 노드 자원을 관리하는 것이다.

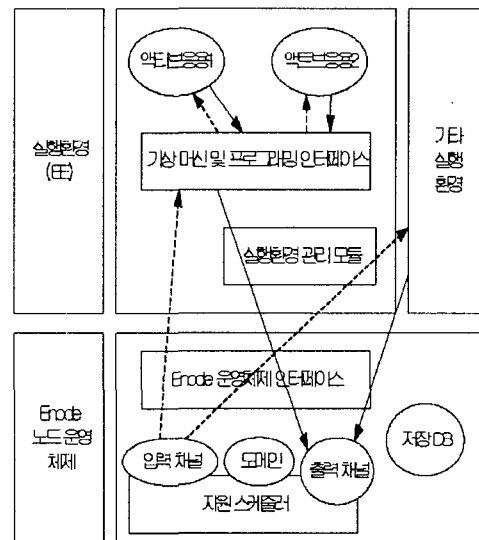


그림 2 Enode 노드 운영체제 구조 (실선 : 입력 메시지, 점선 : 반환메시지)  
Fig. 2 Enode Node Operating System Architecture

그림 2에서 하부 구조에 하드웨어가 존재하고, 이 위에 기존의 운영체제가 있다. 이 운영체제를 이용하여 Enode 운영체제를 설계한다. Enode 운영체제는 도메인 모듈, 채널 모듈 등으로 구성되어 있다. Enode 운영체제 위에 "실행 환경"이 동작한다. 실행 환경에는 여러가지 액티브 응용 프로그램이 동작한다.

논문에서 제안하는 능동 네트워크 Enode 노드 운영체제의 구조는 크게 다섯 부분으로 구성된다: 도메인(domain) 관리, 채널(channel) 관리, 쓰레드(thread) 관리, 메모리(memory) 관리, 파일(file) 관리. 이 구성 요소 간의 논리적인 구조를 그림으로 나타내면 그림 3과 같다.

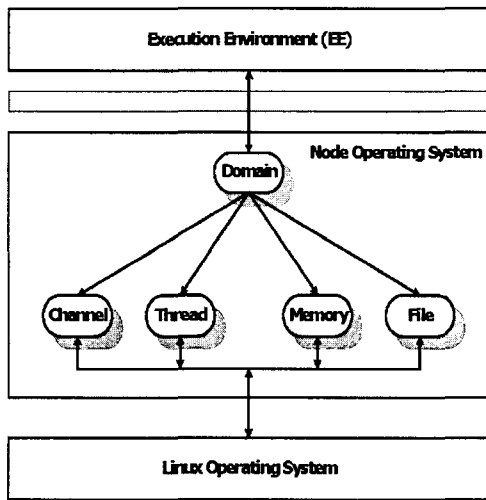


그림 3 Enode 노드 운영체제 구조  
Fig. 3 Enode Node Operating System

도메인(domain) 관리는 회계(accounting), 제어 관리(admission control), 스케줄링 등을 수행하는 모듈로써 "실행 환경"과의 인터페이스를 담당하는 모듈이다. 각 도메인은 특수한 패킷을 실어나르는데 필요한 자원(받고 보내는 메시지, 메모리 풀, 쓰레드 풀 등)을 가지고 있다. 입력 채널(input channel)에 도착한 능동 패킷은 도메인에 할당된 메모리와 쓰레드를 사용하여 "실행 환경"에 의하여 처리된다. 그리고 패킷은 출력 채널(output channel)로 전송된다. 그리고 도메인은 계층적으

로 구성이 가능하도록 설계되어 있다.

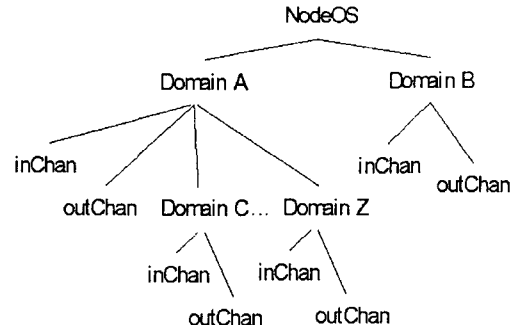


그림 4 도메인의 계층적인 구조  
Fig. 4 Domain Structure

도메인은 패킷을 주고 받기 위해서 채널을 생성한다. 어떤 채널은 "실행 환경"에서 핵심 역할을 수행한다. 이 채널은 "실행 환경"과 하부 구조 사이에 패킷을 전달하는 역할을 수행한다. Enode 노드 운영체제에서 채널은 입력 채널과 출력 채널 역할을 수행한다. 채널은 능동 네트워크의 Enode 운영체제에서 중심 역할을 수행한다. 컷 채널(cut channel)은 "실행 환경"에 의하여 처리되거나 방해되어지지 않는 패킷으로, 이 패킷이 입력 디바이스에서 출력 디바이스로 전송이 된다.

능동 네트워크에서 계산을 위한 기초 모듈로 쓰레드 모듈을 사용한다. 각 도메인은 도메인이 생성될 때 하나의 쓰레드 풀을 갖는다. 쓰레드 풀은 도메인이 생성되면서 초기화된다 [13, 14, 15, 16, 17, 18].

능동 네트워크에서 메모리 모듈은 패킷 버퍼로 사용되어 지고 "실행 환경"의 상태를 저장하는데 이용된다. 메모리 모듈은 메모리를 메모리 풀로 구성하여 사용이 편리하도록 하고 있다. 메모리 풀은 콜백(callback) 함수를 가지고 있다. 이 콜백 함수는 자원의 한계를 초과하게 되면 Enode 노드 운영체제에서 이 함수를 수행하게 된다.

또한 Enode 노드 운영체제가 자원의 한계를 초과하게 될 경우 "실행 환경"에서 이러한 상황이 처리될 수 있도록 요구한다. 파일 모듈은 POSIX 1003.1 인터페이스와 파일 및 디렉토리 간에 계층적인 구조를 지원한다. 본 논문에서 제안하는

Enode 노드 운영체제는 리눅스 운영체제 위에서 동작을 한다. 기존의 리눅스 운영체제의 커널 기능을 이용하여 능동 네트워크의 기능을 제공하기 위한 Enode 노드 운영체제로써의 역할을 수행한다. Enode 노드 운영체제의 핵심 부분은 도메인 모듈이다.

도메인 모듈은 "실행 환경"과 인터페이스를 할 뿐만 아니라 하부의 자원들을 관리하기 위한 주체적인 역할을 수행한다. 도메인은 자원 관리를 위한 4개의 추상적인 관리 모듈과 상호 작용한다. 4개의 추상적인 모듈과 도메인 모듈과의 관계는 다음 그림 5과 같이 표시할 수 있다. 또한, 그림 5은 채널과 다른 자원 관리 모듈로 설계되어 있다.

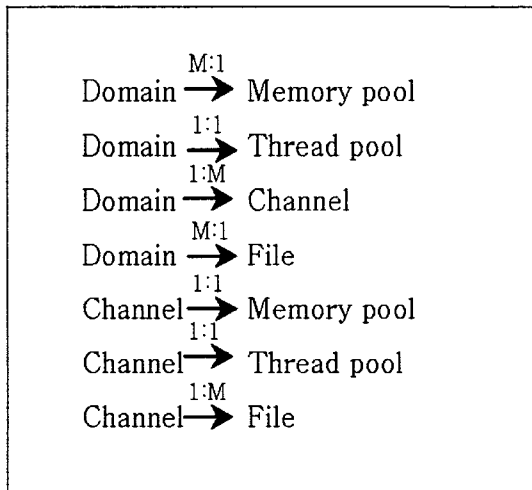


그림 5 도메인, 채널과 다른 모듈과의 관계  
Fig. 5 Relationship among domain, channel and other module

그림 5는 Enode 노드 운영체제에서 도메인, 채널 및 다른 모듈과의 관계를 나타낸다. 그림 5에서 여러 개의 도메인이 메모리 풀, 파일과 연결되어 동작된다. 도메인과 쓰레드는 상호 연동 되어 동작이 된다. 도메인과 여러 개의 채널이 연결되어 동작된다. 채널과 메모리, 쓰레드 풀은 상호 연동하여 동작된다. 채널은 여러 개의 파일을 사용할 수 있다.

### III. Enode 노드 운영체제 설계

Enode 노드 운영체제는 외부 인터페이스로 "실행 환경"과 내부 모듈 상호 간에 그림 5과 같이 데이터를 주고 받는다. Enode 운영체제 내의 각 모듈은 메시지를 상호 주고 받는다. 또한 Enode 운영체제는 가장 상위에 도메인 모듈이 있고, 이 모듈을 지원하기 위한 채널 모듈, 쓰레드 모듈, 메모리 모듈, 파일 모듈로 설계되어 있다.

#### 1. 도메인 모듈(Domain Module)

도메인 모듈은 Enode 노드 운영체제에서 핵심적인 역할을 수행한다. 도메인 모듈은 "실행 환경"에서 처리해 주기를 바라는 요구사항을 받아서 수행한다. 또한 도메인 모듈은 Enode 노드 운영체제의 각 모듈과 상호 연동해서 동작한다. 도메인과 "실행 환경"과의 인터페이스를 위한 자료 구조는 다음 그림 6과 같다.

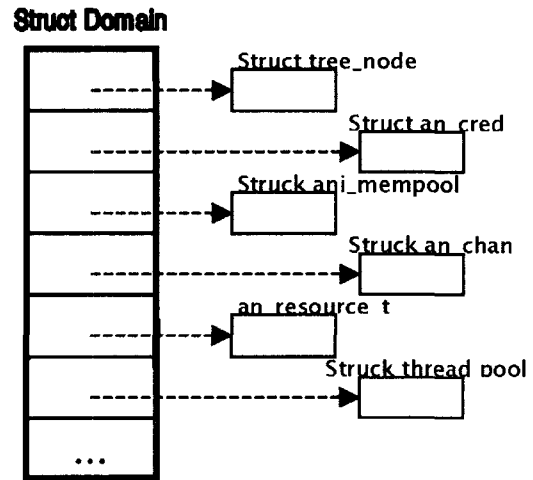


그림 6 도메인 자료 구조  
Fig. 6 Domain Data Structure

domain 구조체는 외부 "실행 환경"과의 도메인 관련 인터페이스를 위한 자료 구조이다. Domain 구조체는 그림 6과 같이 도메인 간의 상호 관계를 정의하는 트리 구조체, 각 도메인에 대한 사용자 접근 권한을 통제하는 구조체, 노드 운영체제 내

부에서 메모리를 관리하는 자료 구조, 채널 관련 자료구조에 대한 정보, 쓰레드와 관련한 풀에 대한 정보를 담고 있다.

## 2. 채널 모듈(Channel Module)

채널 모듈은 도메인 모듈과 외부 네트워크 시스템 간의 패킷을 수신하고 전송하는 역할을 수행한다. 채널은 처리되는 패킷에 따라서 세가지 형태의 채널로 나누어진다.: inchan, outchan, cutchan 이 있다. inchan을 생성할 경우 domain은 다음을 정의해야 한다.

- (1) 도착하는 패킷이 이 채널에 전달되어야 한다.
- (2) 버퍼 풀에서 기다리는 패킷이 이 패킷에 의하여 처리되어야 한다.
- (3) 패킷을 다루기 위한 함수를 정의한다.

-패킷의 전송 내용은

- (1) 프로토콜 spec
- (2) address spec string
- (3) demux key

- 버퍼 풀은 도메인 메모리 풀에서 생성된다.
- 패킷 핸들러는 패킷 전송을 다루고, 도메인 쓰레드 풀 컨텍스트에서 수행한다.

outchan을 생성할 경우 domain은 다음을 정의해야 한다.

- (1) 패킷을 어디로 전달할 것인지 ?
- (2) 채널이 소모하게되는 링크 대역폭이 어느정도 인지?

cutthrough 채널은 receive와 transmit를 동시에 가지고 있다. Cutchan은 이미 만들어져 있는 inchan과 outchan의 연결을 통해서 만들어진다. "실행 환경"에서 cutthrough 채널을 생성한다.

[그림 7]은 "실행 환경"에서 노드 운영체제에 요구하는 채널과 관련한 인터페이스를 위한 자료 구조이다. 이 자료구조는 노드 운영체제 내부에서 관리하는 구조체와 nodeos 개체를 가르키는 포인터로 구성된다.

터로 구성된다.

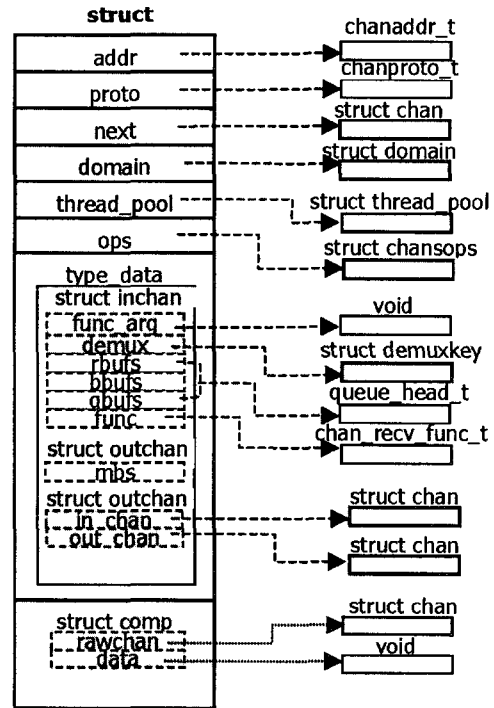


그림 7 채널 자료 구조  
Fig. 7 Channel Data Structure

그림 7에서 addr은 이터넷과 관련한 주소 정보를 가지고 있는 자료 구조이다. proto는 프로토콜의 정보를 담고 있는 자료 구조이다. next는 여러 개의 채널을 링크드 구조체로 연결한다. Domain은 도메인 자료구조를 연결하는 구조체이다. Thread\_pool은 쓰레드 풀의 구조체를 나타낸다. Inchan, outchan, cutchan은 inchan/outchan/cutchan과 관련한 정보를 저장하는 자료 구조이다. 채널과 관련한 그림 7 자료 구조는 공유 영역으로 설정되어 있어서 inchan, outchan, cutchan이 공용해서 사용하는 구조로 되어 있다.

## 3. 쓰레드 모듈(Thread Module)

노드 운영체제에서 쓰레드는 도메인과 채널의 보조 역할을 수행한다. 쓰레드는 계산을 위한 환경을 제공하는 모듈이다. 각 도메인은 도메인이

생성이 될 때 하나의 스레드 풀(single thread pool)을 갖는다. 스레드 풀은 도메인이 생성 될 때 초기화된다.

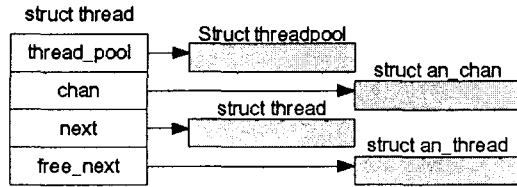


그림 8 스레드 자료 구조  
Fig. 8 Thread Data Structu

thread 구조체에서 thread\_pool은threadpool\_t 구조체를 나타낸다. 이 구조체는 스레드 풀을 관리하는 자료 구조이다. Chan은 채널 자료구조를 나타낸다. Next는 스레드 구조체를 링크드 리스트로 구성하는 연결 정보이다. Free\_next는 스레드들 중에서 free 된 스레드들을 연결하는 정보를 가지고 있다.

#### 4. 메모리 모듈(Memory Module)

Enode 노드 운영체제에서 메모리 관리 모듈은 도메인이나 채널이 수행하는데 필요한 메모리 관리를 주로 담당한다. 메모리 관리 모듈은 패킷이 수행하는데 필요한 정보를 저장하고 "실행 환경"의 상태를 저장하는 역할을 수행한다. Enode 노드 운영체제에서 메모리 관리는 메모리 풀 형식으로 이루어진다. 메모리 풀은 콜 백(call back) 함수를 가지고 있다. 콜 백 함수는 다음의 경우에 기능을 수행한다.

- 자원의 상한 값을 초과하게 되면 Enode 노드 운영체제는 콜 백 함수를 호출
- Enode 노드 운영체제가 풀의 자원 상한 값을 초과하게 될 경우 EE에게 현재 상황을 처리할 수 있도록 요구
- 메모리 풀 계층 구조는 풀들 간에 메모리 공유를 허락
- 스레드들은 메모리 풀에 접근을 mmap-style를 사용할 수 있도록 한다.

Memory사용이 끝나면 unmap 연산을 사용

노드 운영체제는 메모리 할당을 malloc style 인터페이스 형식으로 제공한다.

#### 5. 파일

Enode 노드 운영체제에서 파일 기능은 "실행 파일"에서 필요한 파일과 관련한 기능들을 제공한다. Enode 노드 운영체제에서 파일 기능은 공유 데이터를 관리하는 역할을 주로 한다. Enode 노드 운영체제에서 파일 지원 기능은 POSIX 1003.1 인터페이스를 지원한다. 또한 파일 지원 기능은 계층적인 구조를 가지고 있다. 파일 관련 기능을 지원하기 위한 함수로an\_open(), an\_creat(), an\_read(), an\_write(), an\_close(), an\_lseek() 이 있다.

#### IV. 평가

본 논문에서 제안하는 노드 운영체제의 구조에 대한 동작 특성을 간단히 검증하기 위하여 "실행

```

main()
{
    domain_t *domain;
    void *mem;
    an_cred_t *cred;
    an_cpuspec_t *cspec;
    an_memspec_t *vm;
    char *init;
    void *initarg;
    char *exit_func;

    mem = NULL;
    domain = an_domain_create(mem,
        cred, cspec, vm, init, initarg,
        exit_func);
}
    
```

그림 9 실험을 위한 "실행환경" 프로그램  
Fig. 9 "Experimental Environment" Program

환경" 수준에서 도메인의 생성을 통해서 Enode

노드 운영체제의 동작이 정확히 이루어 짐을 보이는 실험을 수행하였다. 실험을 위한 간단한 "실행 환경" 응용 프로그램을 작성하였다. 실험을 위한 "실행 환경" 응용 프로그램은 다음과 같다.

본 논문에서 제안하는 노드 운영체제를 리눅스(Linux) 운영체제에서 구현하여 실험하였다. 그림 9에서 제안하는 "실행 환경" 응용 프로그램을 수행하여 Enode 노드 운영체제의 내부 수행 과정을 검증한다. 위에서 보여주는 "실행 환경" 응용 프로그램을 수행하였을 때 다음 그림 10과 같은 수행 결과를 보여준다.

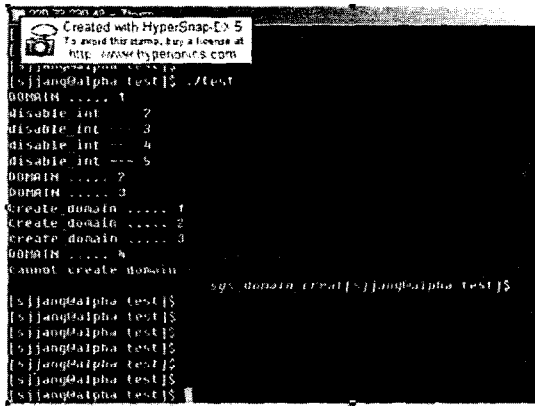


그림 10 Enode 노드 운영체제에서 도메인 생성 과정 점검을 위한 실험  
Fig. 10 Experiment to create the domain in the Enode OS

그림 10은 Enode 노드 운영체제에서 도메인 생성 과정 점검을 위한 실험 결과이다. 이 화면에 나타나는 문자 메시지는 도메인 생성시 호출하게 되는 주요 함수에 "printf" 문장을 삽입하여 출력하도록 만들었다. 실제 도메인 생성을 정상적으로 수행하기 위해서는 쓰레드, 메모리 부분 등이 완전히 프로그램 되어야 한다. 현재 Enode 노드 운영체제 실험 환경에서 구현이 되지 않은 함수는 리턴하도록 하였기 때문에 이런 부분은 정상적인 수행을 할 수가 없다. 따라서 최종 도메인 생성 결과는 도메인을 생성할 수 없다는 메시지("cannot create domain")가 출력이 된다. 이 실험 결과는

Enode 노드 운영체제에서 도메인 생성시에 호출되는 주요 함수를 실제 정상적으로 거치는지를 실험하기 위한 것이다. 따라서 그림 10에서 보는 바와 같이 출력되는 문자 메시지들을 통해서 Enode 노드 운영체제에서 도메인이 정상적으로 처리됨을 알 수 있다.

### V. 결론

본 논문은 능동 네트워크에서 Enode 노드 운영체제를 제안하고 설계하였다. Enode 노드 운영체제는 능동 네트워크에서 핵심적인 역할을 수행하는 모듈이다. 본 논문에서는 능동 네트워크에서 핵심적 역할을 수행하는 Enode 노드 운영체제를 제안한다. 본 논문에서 제안하는 Enode 노드 운영체제는 크게 5개의 모듈로 이루어져 있다. 5개의 모듈은 도메인 모듈, 채널 모듈, 쓰레드 모듈, 메모리 모듈, 파일 모듈로 구성된다. 5개의 모듈 중에서 도메인과 채널이 Enode 노드 운영체제의 핵심적인 역할을 수행한다. 도메인 모듈과 채널 모듈 이외의 쓰레드 모듈, 메모리 모듈, 파일 모듈은 도메인과 채널의 기능을 보조하는 역할을 수행한다. 본 논문에서는 Enode 노드 운영체제에서 도메인과 채널 기능에 대해서 설계 및 구현하였다. Enode 노드 운영체제에서 도메인 모듈은 능동 네트워크에서 도메인을 관리하는 역할을 담당한다. 도메인 모듈은 새로운 도메인을 생성 및 이미 존재하는 도메인을 제거하는 역할을 수행한다. Enode 노드 운영체제에서 도메인 모듈은 "실행 환경"과의 인터페이스 역할을 하고, Enode 노드 운영체제의 각 모듈과 상호 연동해서 동작한다. Enode 노드 운영체제에서 채널 모듈은 inchan, outchan, cutchan으로 구성된다. 채널은 도메인과 외부 네트워크 시스템 간의 패킷을 수신하고 전송하는 역할을 수행한다. 처리되는 패킷에 따라서 inchan, outchan, cutchan의 세가지 형태의 채널로 나뉘어진다. 본 논문에서 제안한 Enode 노드 운영체제의 동작을 점검하기 위하여 "실행 환경"에 응용 프로그램을 만들어 핵심 기능들이 동작이 되는지 실험하였다.

참고 문헌

- [1] 이중수, 이승현, 이영희, "Active Network 구조 : 문제점 및 접근 방법", Sigcomm Review, pp. 109-126, 2000.12.
- [2] 이남희, "Active 네트워크 기술 개발 현황", Sigcomm Review, pp. 127-142, 2000.12.
- [3] Calvert, K. et al, "Architectural Framework for Active Networks", an Working Group Draft, July 1999.
- [4] L. Perterson. Et al, "NodeOS Interface Specification", an Node OS Working Group, Feb. 1999.
- [5] Danny Raz and Yuval Shavitt, "Active Networks for Efficient Distributed Network Management", IEEE Communications Magazine, March 2000.
- [6] 한국전자통신연구원, 차세대인터넷을 위한 능동보안 기술 백서, 한국전자통신연구원, 2001.5.15.
- [7] Active Networks Working Group, "Architectural Framework for Active Networks", July 1998, <http://www.cc.gatech.edu/projects/canes/arch/arch-0-9.ps>.
- [8] AN Node OS Working Group, "NodeOS Interface Specification", <http://www.cs.princeton.edu/nsg/papers/nodeos99.ps>.
- [9] D. Scott Alexander, Marianne Shaw, Scoot M. Nettles, and Jonathan M. Smith. Active bridging. In Proceedings of the ACM SIGCOMM '97 Confrence, pages 101-111, September 1997.
- [10] Gaurav Banga, Peter Druschel, and Jeffrey Mogul. Resource containers: A new facility for resource management in server systems. In Proceedings of the 3rd Symp. On Operation System Design and Impl., pages 45-58, February 1999.
- [11] Samrat Bhattacharjee, Ken Calvert, and Ellen Zegura. Congestion control and caching in CANES. In ICC '98, 1998.
- [12] Michael Hicks, Pankaj Kakkar, Jonathan T. Moore, Carl A. Gunter, and Scott Nettles. PLAN: A packet language for active networks. In ICFP 98, pages 86-93, September 1998.
- [13] IEEE P1520 Working Group. IEEE P1520: Proposed IEEE standard for application programming interfaces for networks web site. <http://www.ieee-pin.org/>.
- [14] K. Calvert, Ed. Architectural Framework for Active Networks. Technical report, AN Architecture Working Group, 2000.
- [15] M. Frans Kaashoek, dawson R. Engler, Gregory R. Ganger, Hector Briceno, Russell Hunt, David Mazieres, Thomas Pinckney, Robert Grimm, John Jannotti, and Kenneth Mackenzie. Application performance and flexibility on exokernel systems. In Proceedings of the 16th ACM Symp. On Operating Systems Principles, pages 52-65, St. Malo, France, October 1997.
- [16] Tal Lavian, Robert Jaeger, and Jeffrey Hollingsworth. Open programmable architecture for java-enabled network devices. In Proc. of the Seventh IEEE Workshop on Hot Interconnects, Stanford University, CA, August 1997.
- [17] S. Murphy, E. Lewis, R. Puga, R. Watson, and R. Yee. String security for active networks. In The Fourth IEEE Conference on Open Architectures and Network Programming, Anchorage, Alaska, April 2001.
- [18] David Clark. The design philosophy of the DARPA Internet protocols. In Proceedings of the SIGCOMM '88 Symposium, pages 106-114, August 1988.
- [19] S. Merugu, S.Bhattacharjee, E.Zegura, K.Calvert, "Bowman : A Node OS for Active Networks", <http://www.cs.princeton.edu/nsg/papers>



저자 소개



**장승주(Jang, Seung Ju)**

1985년 부산대학교 계산통계학과 졸업(계산학)  
1991년 부산대학교 계산통계학과(계산학 이학석사)

1996년 부산대학교 컴퓨터 공학과(공학박사)  
1987년 - 1996년 한국전자통신연구원 시스템 S/W 연구실  
2000-2002 Univ. of Missouri at Kansas City, Visiting Professor  
1996년 - 현재 동의대학교 컴퓨터공학과 부교수  
e-mail: sjjang@hyomin.dongueui.ac.kr  
※ 관심 분야 : 운영체제, 보안, 분산처리 시스템



**나중찬(Na, Jong Chan)**

1986년충남대학교 계산통계학과 학사  
1989년승실대학교 전산학과 석사

1989년~현재 한국전자통신연구원 선임연구원  
e-mail : njc@etri.re.kr  
※ 관심분야 : 실시간 시스템, 정보 보호, 인터넷 기반 S/W



**이영석(Lee, Young Suk)**

1992년 충남대학교 컴퓨터 공학과 학사  
1994년 충남대학교 컴퓨터 공학과 공학석사

2002년 충남대학교 컴퓨터공학과 공학박사  
1994년~1997년 LG전자 연구원  
2002년 ~ 현재 한국전자통신연구원 선임연구원  
e-mail : ysllee@etri.re.kr  
※ 관심분야 : 분산시스템, 정보 보호, 이동컴퓨팅