액티브 네트워크 환경에서 대응 메커니즘을 이용한 노드 생존성에 관한 연구

양 진 석[†]·이 호 재[††]·장 범 환[†††]·김 현 구[†]·한 영 주[†]·정 태 명[††††]

요 약

방화벽이나 침입 탐지 시스템 같은 기존의 보안 솔루션들은 새로운 공격에 대한 탐지 오보율이 크고 내부 공격자의 경우 차단할 수 없는 등 여러 가지 단점이 있다. 이러한 보안 솔루션의 단점은 시스템의 가용성을 보장하는 메커니즘으로부터 보완할 수 있다. 노드의 생존성을 보장하는 메커니즘은 여러 가지가 있으며, 본 논문에서는 실시간 대응 메커니즘을 이용한 침입 감내(intrusion tolerance)를 접근 방법으로 한다. 본 논문에서 제시하는 생존성은 관심을 가지는 시스템 자원을 모니터링하고 자원이 임계치를 초과하면 모니터링 코드 및 대응 코드를 액티브 네트워크 환경에서 자동적으로 배포하여 동작하게 함으로써 시스템의 가용성을 능동적으로 보장하는 메커니즘을 제시한다. 자원 모니터링은 본 논문에서 제안한 평균 프로세스에 기반한 동적인 자원 제어 기법을 통해 수행한다. 대응 코드는 노드의 가용성을 위해 액티브 노드에 상주하거나 요청이 있을 때 해당 작업을 수행한다. 본 논문은 기존의 보안 솔루션이 갖는 단점에 대한 고찰을 통해 이를 보완한 침입 감내 메커니즘을 제시하고, 시스템 재설정 및 패치 수동성에 대한 단점을 액티브 네트워크 기반구조가 제공하는 서비스의 자동화된 배포 등의 장점을 통합한 노드의 생존성 메커니즘을 제시한다.

A Study on Survivability of Node using Response Mechanism in Active Network Environment

Jin-Seok Yang[†]·Ho-Jae Lee[††]·Beom-Hwan Chang[†††]
Hyoun-Ku Kim[†]·Young-Ju Han[†]·Tai-Myoung Chung[††††]

ABSTRACT

Existing security solutions such as Firewall and IDS (Intrusion Detection System) have a trouble in getting accurate detection rate about new attack and can not block interior attack. That is, existing security solutions have various shortcomings. Shortcomings of these security solutions can be supplemented with mechanism which guarantees an availability of systems. The mechanism which guarantees the survivability of node is various, we approach intrusion tolerance using real time response mechanism. The monitoring code monitors related resources of system for survivability of vulnerable system continuously. When related resources exceed threshold, monitoring and response code is deployed to run. These mechanism guarantees the availability of system. We propose control method about resource monitoring. The monitoring code operates with this method. The response code may be resident in active node for availability or execute a job when a request is occurred. We suggest the node survivability mechanism that integrates the intrusion tolerance mechanism that complements the problems of existing security solutions. The mechanism takes advantage of the automated service distribution supported by Active Network infrastructure instead of passive system reconfiguration and patch.

키워드 : 액티브 네트워크(Active Network), 침입감내(Intrusion Tolerance), 생존성(Survivability)

1. 서 론

최근 발생하는 컴퓨터 시스템에 대한 공격은 자동화 및 분산화 되고 있다. 이로 인한 피해 규모가 상당히 크며 피해 정도도 심각하다. 이렇게 큰 피해가 발생하는 원인은 네트워크 및 시스템 관리자의 보안 의식 결여가 근본적인 원인이지만 공격의 자동화에 비해 시스템 설정, 보안 패치 등은 수동적인 방법이라는 것도 원인이 된다.

2003년 1·25 인터넷 대란은 관리자의 보안 의식 결여와 대응 체계의 미비뿐만 아니라 효율적이고 자동적인 대응 방법의 부재에서 국가적으로 인터넷이 마비되는 큰 사고로 이어졌다. 이러한 대란의 원인인 SQL Slammer 웜은 이미 취약성이 발표되었지만, 해당 네트워크 관리자나 시스템 관

† 준 회 원 : 성균관대학교 대학원 컴퓨터공학과
†† 정 회 원 : 한국정보보호진흥원 연구원
††† 준 회 원 : 한국전자통신연구원 연구원
†††† 종신회원 : 성균관대학교 전기전자 및 컴퓨터공학부 부교수
논문접수 : 2003년 7월 22일, 심사완료 : 2003년 9월 24일

리자의 보안 의식 부재에서 비롯되었고 해당 취약성에 대한 보안 패치나 재설정 등은 여전히 수동적으로 이루어지고 있다[4, 21]. 보안 패치나 재설정은 믿을만한 솔루션이지만 항상 적절한 시기에 수행하는 것이 가능하지 않다[7, 17]. 이러한 수동적인 보호 방법은 1·25 인터넷 대란의 피해를 더욱 크게 하였다. 많은 보안 관련 업체 및 기관은 이러한 공격에 대한 방어를 위해 많은 노력을 하고 있지만 새로운 형태의 공격에 대해서는 속수무책이었다. 회사나 조직에서 사용하는 보안 솔루션들 중 대표적으로 침입 탐지 시스템과 침입 차단 시스템이 있는데 이런 시스템 역시 무용지물이었다. 대표적인 보안 솔루션인 침입 탐지 시스템과 차단 시스템의 단점을 살펴보면 다음과 같다.

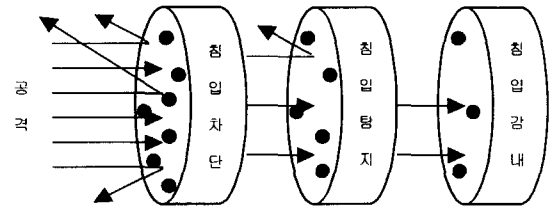
- 침입 탐지 시스템
  - 오탐지(false positive)
  - 미탐지(miss detection)
  - 네트워크 침입탐지시스템의 실시간 탐지 불가능
- 침입 차단 시스템
  - 공격 트래픽과 정상 트래픽의 구분이 어려움
  - 네트워크 내부의 악의적인 사용자에 대해서는 방어를 할 수 없음

침입 탐지 시스템의 경우, 공격은 다양해지고 있는 반면에 탐지 능력은 제한되어 있으며 오탐율을 줄이기 위한 많은 연구가 있지만 기존의 침입 탐지 알고리즘으로는 세션 기반 탐지의 오탐율이 상당히 높으며, 또한 알려지지 않은 공격에 대한 패턴이 없으면 분석 및 탐지가 어렵다. 이러한 단점은 비정상 탐지 기법을 이용하여 보완하고 있으나 현재까지는 공격과 비 공격을 분별하는데 한계가 있고 오탐율을 높일 수 있다는 단점을 가지고 있다. 네트워크 기반의 침입 탐지 시스템은 네트워크 상의 패킷을 탐지하지만 대부분의 패킷은 네트워크 에이전트가 패킷을 분석하고 판정하기 이전에 침입이 가능한 단점을 가지고 있다[14, 23].

침입 차단 시스템의 경우, IP 주소와 TCP/UDP 포트번호를 기반으로 동작하므로 허용된 IP 주소나 TCP /UDP 포트 번호로 들어오는 트래픽의 경우 모두 내부 네트워크로 포워딩한다. 분산 서비스 공격과 같은 경우, IP 속이기(spoofing)를 이용해 정당한 IP 주소를 가지고 공격을 하기 때문에 정상 트래픽과 구분이 어렵다. 또한, 침입 차단 시스템 내부의 악의적인 공격자에 대해서는 방어할 수 없는 단점을 가지고 있다[22].

이러한 단점들은 생존성에 대한 연구로 발전되었다. 고장 감내(fault tolerance)나 침입 감내(intrusion tolerance)는 기존의 보안 솔루션의 단점을 보완하기 위한 생존성에 대한 연구들이다[10, 24]. 이러한 연구의 목표는 공격과 시스템 결함으로 피해가 발생하더라도 필수적인 서비스를 지속적으로 제공하고자 하는데 있다[24].

(그림 1)은 침입 차단과 탐지, 감내 시스템의 구조를 보여준다. 아무리 보안이 잘되어 있는 시스템이라 할지라도 모든 공격에 대한 방어를 하는 시스템은 없다. (그림 1)은 침입 차단 시스템과 침입 탐지 시스템이 방어하지 못한 공격에 대해서 침입 감내 시스템의 역할을 구조적으로 보여준다. 침입 감내는 피해복구 기술 중의 하나로써 시스템에 여러 가지 보호 및 대응 방법으로도 미처 대처하지 못한 공격이나 침입에 대한 피해를 신속히 복구하여 시스템의 가용성을 제공하여 주는 기술이다[24].



(그림 1) 침입 차단, 탐지, 감내 시스템 구조

본 논문에서 기술하게될 감내 메커니즘은 필수 서비스가 할당받고 있는 자원을 모니터링하고 적절한 대응을 통해 시스템의 가용성을 보장한다. 이러한 침입 감내 메커니즘은 액티브 네트워크를 기반 구조로 하며 기존의 수동적인 방어 및 대응을 자동화하여 노드의 생존성을 가능하게 한다.

본 논문의 구성은 다음과 같다. 2장에서는 액티브 네트워크와 침입 감내에 대한 내용을 기술하고, 3장에서는 침입 감내 시스템이 액티브 네트워크 기반 구조에서 갖는 장점을 기술하며, 4장에서는 감내 메커니즘을 액티브 네트워크 기반 구조에 적용할 때 고려해야할 사항에 대해서 설명한다. 5장에서는 액티브 네트워크 기반구조에서 침입 감내를 위한 대응 기술과 자원 모니터링 알고리즘을 기술하며 6장에서 동작 시나리오를 기술한다.

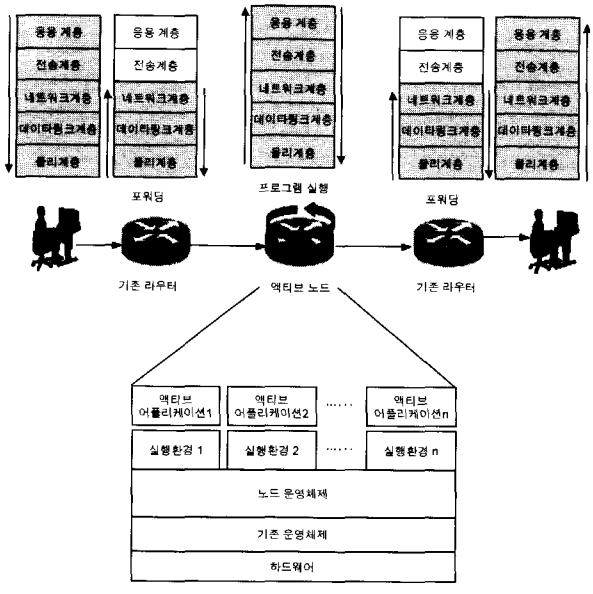
2. 관련 연구

본 장에서는 기반 구조가 되는 액티브 네트워크에 대한 개념을 통해 액티브 네트워크가 어떻게 자동화된 대응 구조를 갖는지에 대하여 기술한다. 또한, 본 논문에서 제안하는 침입 감내에 대한 개념과 최근 연구 경향에 대해서 기술한다.

2.1 액티브 네트워크

액티브 네트워크는 기존 네트워크의 문제점을 보완하고자 중간 노드에서 사용자 프로그램이 실행 가능하게 한다. [6]은 기존 네트워크의 문제점을 기술하였다 : 첫째, 공유된 네트워크 자원에 새로운 기술과 표준을 통합하는데 따르는

어려움 ; 둘째, 몇몇 프로토콜 계층의 과도한 운용으로 인한 성능 문제 ; 셋째, 기존의 아키텍처 모델에 새로운 서비스를 제공하는데 따르는 어려움을 지적하였다. 액티브 네트워크는 이러한 문제점을 보완하고자 중간 노드를 실행 가능하게 만들어 현재 중간노드들의 "저장-전달"의 기능에서 "저장-처리-전달"의 기능을 갖게 하여 유연하고 동적인 네트워크 구조를 제공한다[1-3]. (그림 2)는 액티브 네트워크의 개념을 나타낸다. 액티브 노드는 들어오는 패킷을 수신하여 실행 여부를 결정하고 실행 시에는 실행환경으로 액티브 코드를 전달하여 실행한다. 액티브 노드는 실행 결과에 따라 수신한 코드를 그대로 보내거나 새로운 코드를 가진 패킷을 생성하여 다음 노드로 전달한다. 중간 노드의 실행은 노드 운영체제와 실행환경을 구축함으로써 가능하다.



(그림 2) 액티브 네트워크 개념

노드운영체제는 노드의 통신, 메모리, 노드에서 진행하는 각종 패킷의 흐름 사이에 자원을 관리하는 등의 역할을 수행한다. 액티브 네트워크 노드운영체제 워킹 그룹에서는 노드운영체제의 원활한 수행을 위해 도메인, 쓰레드 풀, 메모리 풀, 채널, 파일에 대한 추상적 개념을 참고문헌[1, 2]에 정의하고 있다.

실행환경은 액티브 어플리케이션의 관리 및 지원을 위한 프로그래밍 모델을 정의한다. 액티브 네트워크 백본(ABone)에서 현재 운용 및 연구중인 실행환경은 ASP(Active Signaling Protocol), ANTS(Active Node Transfer System), PLAN(Packet Language for Active Network)이다. 실행환경으로 사용되는 언어는네트워크 내에서 코드가 여러 머신으로 이동한다는 측면에서 보안, 안정성, 에러회복, 성능, 플랫폼 독립성, 객체 직렬화, 동적으로 새로운 기능을 추가할 수 있는 소프트웨어 이동성 등의 특징들이 요구된다[5, 11, 15, 19]. 각각의 실행환경은 이러한 특징을 만족하는 Java나 Caml 등의 기존의 프로그래밍 언어를 사용하거나 PLAN이라는 액티브 네트워크 환경에 최적화된 언어를 새롭게 만들어 사용한다[5, 11, 15].

2.2 침입 감내

생존성을 위한 피해 복구 기술은 시스템의 여러 가지 보호 방법이나 대응 방법이 적용되었다 할지라도 미처 대응하지 못하는 침입이나 공격으로 인한 시스템의 가용성을 보장하는 기술이다[10, 13, 14, 24]. 침입 감내는 이러한 피해 복구기술 방법 중에 하나이다. 결함 감내 시스템의 개념이 적용된 침입 감내의 대표적인 예는 복제(replication) 시스템이 있다[10, 24]. 이 기술은 서비스를 제공하는 시스템에 대해 다중의 복사본을 분산하여 배치시켜 하나의 노드가 침입에 의해 다운되거나 서비스가 중단되어도 복사본을 이용하여 서비스를 지속적으로 제공하는 기술이다[24]. 최근에는 침입 감내 연구 중 증상(symptom) 참조를 위해 시스템 자원 모니터링을 통한 적응성 있는(adaptive) 어플리케이션에 대한 연구가 진행되고 있다[10, 13, 14, 24].

3. 동 기

최근의 공격 경향은 웜과 스크립트 코드 등을 이용한 자동화된 공격 방법을 사용하고 있다. 반면에 현재의 방어를 위한 솔루션들은 방화벽 룰의 변경, 시스템 재설정, 보안 패치와 같은 수동적인 대응에 의존한다. 현재 동향은 자동화된 공격으로 인해 시스템 패치 및 보안성을 높이기 위해 관리자에게 주어진 시간이 빠르게 줄어들고 있다[16,17].

시스템 재설정이나 패치는 믿을만한 솔루션이지만 항상 적절한 시기에 적용하는 것이 가능하지 않을 뿐만 아니라 보안에 대한 수동적 대응만을 제공하는 네트워크 기반 구조로 인해 침입에 대한 피해는 더 클 수밖에 없다[4, 7, 17]. 많은 보안 관련 업체 및 연구 기관은 이러한 공격에 대한 방어를 위해 많은 노력을 하고 있지만 네트워크 기반 구조의 비유연성과 각종 보안 솔루션의 단점들로 인해 기대만큼의 성과를 얻지 못하고 있다. 현재까지 보안 솔루션의 단점으로 인해 생존성을 위한 연구가 시작되었는데 그 중 하나가 침입 감내 기술이다. 침입 감내 기술은 기존의 보안 솔루션이 차단 및 탐지하지 못한 침입이 있을 것이라는 가정 하에 기존의 서비스를 지속적으로 제공하고자 하는데 목표를 가지고 있다. 본 논문의 침입 감내에 대한 접근 방법은 자원 모니터링을 통한 실시간 대응을 통해 노드의 생존성을 보장하고자 한다.

액티브 네트워크는 취약성 분석, 방어, 대응에 있어서 기존의 네트워크 기반구조에서 제공하지 못하는 다음과 같은

장점들을 가진다[6, 8, 18, 19].

- 새로운 서비스의 빠른 배포
- 새로운 서비스의 자동화된 배포
- 제 3자(third party)에 의한 배포
- 중단없는 실행(non-disruptive)
- 확장성(scalability)
- 적응성(adaptiveness)

액티브 네트워크의 장점은 자동화된 공격에 대하여 자동화된 방어 및 대응 코드를 배포할 수 있는 기반구조를 제공함으로써 기존의 방어 및 대응 체계의 단점을 극복할 수 있다. 침입 감내를 수행하는 코드를 액티브 네트워크 기반구조에서 자동적으로 배포함으로써 기존의 보안 솔루션의 단점과 네트워크 기반구조의 단점을 모두 극복할 수 있다.

4. 고려 사항

액티브 네트워크는 중간노드를 실행 가능하게 만드는 기반 구조를 제공하는데 본 논문에서는 종단 노드도 실행 가능하다는 가정을 한다.

기존의 침입 감내를 수행하는 코드들은 종단 노드에서의 수행이 전제이지만 액티브 네트워크 환경에서는 중간 노드에서의 수행도 가능하므로 종단 노드 및 중간 노드의 특성을 고려하여야 한다. 특히, 액티브 네트워크의 특성상 크게 두 가지 측면에서 성능 문제를 고려해야 한다.

첫째, 코드가 이동한다는 측면에서 네트워크의 성능 문제
둘째, 중간 노드가 패킷 처리뿐만 아니라 노드운영체제와 실행환경이 실행 중이라는 측면에서 노드의 성능 문제

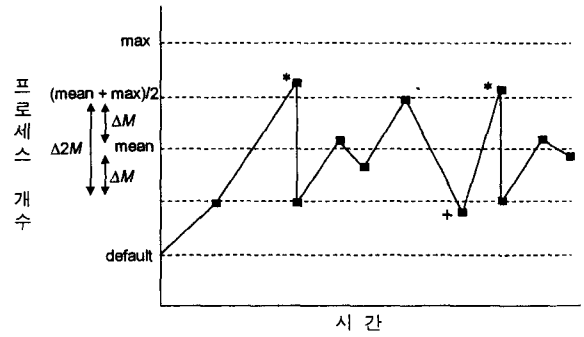
따라서, 본 논문에서는 생존성에 대한 기술 중 복제 기술에 대한 것은 배제하였다. 또한, 대응 코드에 대한 기능은 새로운 프로그램을 인스톨하지 않고 운영체제에서 제공하는 유틸리티를 사용하여 위에서 기술한 두 가지 성능 문제에 대하여 보완하였다.

5. 노드 생존성 기법

본 장에서는 기본적으로 포함되어 있어야 하는 생존성을 위한 기능들을 기술하고 각 기능들의 특징에 따라 종단 노드에서 가능한 기능과 중간 노드에서 가능한 기능으로 분류하였으며, 각 기능이 동작하기 위한 알고리즘을 기술한다.

5.1 모니터링 기법

서비스를 제공하기 위한 시스템의 기본 단위는 프로세스이며 프로세스를 제한하는 메커니즘은 현재 대부분의 운영체제에도 존재한다[9, 20, 25]. 그러나, 이 메커니즘은 정적이며, 본 논문에서는 모니터링을 통한 이상 증상 참조를 위해 동적인 알고리즘을 제안한다.



(그림 3) 증상 참조 모니터링 알고리즘

(그림 3)에서 보는 바와 같이 *default* 프로세스의 개수는 서비스(혹은 데몬)가 처음 시작될 때 기본적으로 실행되는 프로세스의 개수이다. *mean*은 평균 프로세스의 개수이다. 평균 프로세스의 개수(M)는 식 (1)과 같이 계산된다.

$$M = \alpha * M_{new} + (1-\alpha) * M_{old} \quad (1)$$

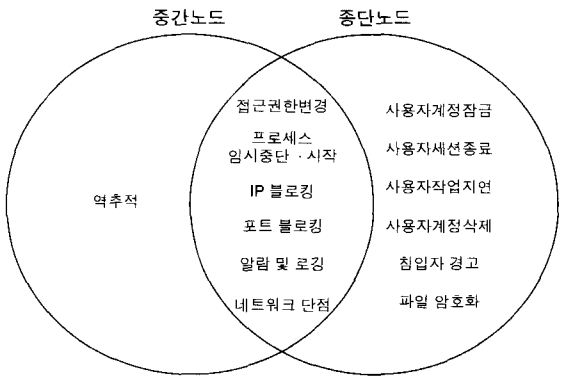
α는 업데이트 계수를 의미하고, M_{new}는 최근의 평균 프로세스 개수를 나타낸다. M_{old}는 이전의 평균 프로세스 개수이고, 평균 프로세스 개수의 값을 계산할 때 업데이트 계수(α)가 0.5 이상이면 최근 평균 프로세스 개수에 가중치를 두는 것이다. *max*는 식 (2)와 같이 계산된다.

$$max = \frac{\text{운영체제의 최대프로세스의 수}}{\text{현재 서버데몬의 수}} \quad (2)$$

여기서, *mean*과 *max*의 중간 값을 구할 수 있다. 중간 값($\frac{mean+max}{2} = risingM$)은 평균 프로세스 값(*mean*)과의 거리(ΔM)를 계산하여 *mean*을 기준으로 아래로 같은 거리의 점(*fallingM*)을 잡는다. 총 거리는 $\Delta 2M$이 되고 프로세스 개수는 $\Delta 2M$ 사이에 위치하여야 한다. 만약, *fallingM* 밑으로 떨어지면((그림 3)에서 +의 경우) 로깅 및 알람 등의 기능이 동작하고 *fallingM*에서 *risingM*까지 한 번에 프로세스의 개수가 늘어나면((그림 3)에서 *의 경우) 적절한 대응 기능이 수행된다.

5.2 대응 기법

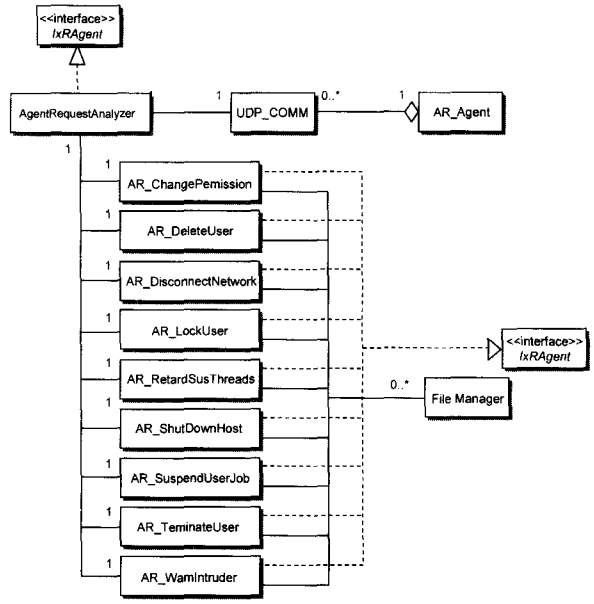
대응 메커니즘을 이용한 노드의 생존성 확보를 위한 기능은 (그림 4)에서 보는 바와 같이 중간노드에서 필요로 하는 기능과 종단 노드에서 필요로 하는 기능으로 구분될 수 있다.



(그림 4) 대응 기능 분류

각각의 대응 기술들은 <표 1>에서 보는 바와 같다. 대응 레벨의 경우 분류 기준은 대응 기능 동작 후 자원 보유상태를 기준으로 상·중·하 세 개의 레벨로 나누었다.

(그림 5)는 대응 기능 테스트를 위해 구현된 코드의 클래스 다이어그램(class diagram)이다. *UDP_COMM*은



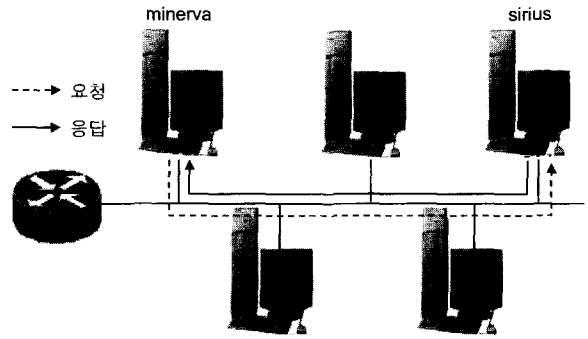
(그림 5) 대응 코드의 클래스 다이어그램

〈표 1〉 대응 기술 및 대응 레벨 분류

대응 기능	설 명	시스템 지원 명령어	대응 레벨
사용자 계정 잠금	사용자 세션 종료 기능과 연관된 기능으로써 시스템 자원의 과도한 사용으로 인해 경고를 발생한 사용자의 계정을 잠그는 기능이다. 관리자는 위험한 사용자로 판단된 경우 단순히 계정을 잠그는 기능과 사용자 계정 삭제 기능과 연관되어 레벨별로 대응할 수 있다.	passwd -l [계정]	하
사용자 세션 종료	특정 사용자가 많은 자원을 점유하고 있을 때 필요한 대응 기능이다. 특정 사용자가 임계치 이상의 시스템 자원을 사용할 경우 해당 사용자의 세션을 종료할 수 있다.	pkill -9 -u [계정]	중
사용자 작업 지연	인증을 받은 사용자임에도 불구하고 사용자의 실수로 과도한 자원을 사용하는 경우가 있을 수 있으므로 필요하다. 이러한 경우 사용자는 시스템의 많은 자원을 점유하게 되는데 이를 방지하기 위한 기능이다. 실제로 이러한 사용자의 실수는 시스템 붕괴로까지 이어질 수 있다. 사용자 작업 지연 대응은 많은 자원을 소비하는 프로세스를 구별하여 해당 프로세스의 우선 순위를 낮추는 방법이다.	nice -[순위] [프로세스 아이디]	하
사용자 계정 삭제	임계치 이상의 경고가 누적된 사용자에 대해서 계정을 삭제하는 기능이다. 이 기능은 레벨별로 계정 삭제와 홈 디렉토리까지 삭제하는 대응이 있을 수 있다.	userdel [계정] userdel -r [계정]	중
접근 권한 변경	접근 권한 변경 기능은 파일 혹은 디렉토리의 접근 권한 변경을 말한다.	chmod [접근권한] [파일,디렉토리]	하
침입자 경고	대부분의 공격자들은 자신들이 모니터링 되고 있음을 인식하지 못하거나 침입 탐지 시스템을 피할 수 있다고 생각한다. 따라서, 침입에 대한 경고 메시지도 일종의 대응 수단이 될 수 있다.	wall(rwall) [사용자]	하
프로세스 임시 중단 · 시작	프로세스 임시 중단 · 시작 기능은 사용자 프로세스를 즉시 중단시킨 후 중단되었던 작업을 다시 시작시킬 수 있는 것이 사용자 세션 종료 기능과 다르다.	pstop [프로세스 아이디] prun [프로세스 아이디]	중
IP 블로킹	IBAN에서도 구현되었던 기능으로써 라우팅 테이블을 이용해서 구현할 수 있다[18].	route add -host [IP 주소] reject	상
네트워크 단절	네트워크 인터페이스를 고립시키는 것이다. 엄밀히 말하면 이 방법은 고립시킨 인터페이스로 들어오는 서비스에 대한 가용성을 보장하지 못한다. 그러나, 해당 노드 및 다른 인터페이스의 서비스는 가용성을 유지시킬 수 있다.	ifconfig [인터페이스 이름] down	상
포트 블로킹	여러 다른 노드에서 같은 포트로 많은 요청이 있을 때 IP 블로킹으로는 방어하기 힘들기 때문에 꼭 필요한 기능이다.	—	상
파일 암호화	공격자의 공격을 어렵게 하기 위해 파일 암호화기능은 필요하다. 중간 노드에서 파일 암호화를 하려면 성능에 상당히 심각한 영향이 있으므로 종단 노드에서 중요한 파일에 대한 암호화 기능을 수행하도록 한다.	—	중
역추적	역추적 기능은 시스템이 공격의 근원지를 알아내어 근원지 네트워크에서 해당 IP를 차단 및 대응함으로써 공격을 근원지에서 막을 수 있도록 한다.	—	중

통신 모듈이며 통신 모듈이 받은 메시지는 *AgentRequest Analyzer*로 보낸다. *AgentRequestAnalyzer*는 모니터링 코드로부터 받은 요청을 구별하고 해당 에이전트를 호출한다. 각각의 에이전트는 *FileManager*를 통해 호출시 필요한 정보를 담은 파일에 접근한다.

<표 1>에서 기술한 몇 가지 대응 기법은 (그림 6)에서 보는 바와 같이 기존의 네트워크 환경에서 테스트하였다. 대응 코드 테스트 환경은 유닉스 기반의 minerva 머신에서 *AR_Manager* 테스트 클라이언트가 요청을 하고 sirius에서 수행되어 그 결과 값을 다시 minerva 머신으로 통보한다. 테스트를 위한 *AR_Manager*는 모니터링 코드와 같이 대응 코드를 호출하거나 대응 코드에 입력 값을 제공한다.



(그림 6) 대응 코드 시뮬레이션 환경

```
----------대응 결과----------
능동대응에이전트 ID     : 100
서버측에서 받은 시간    : 20037131340
대응 레벨               : 1
대응결과 메세지         : 요청하신 작업이 성공하였습니다.
------------------------------
[root@minerva:/export/home/jsyang]java AR_Manager 120 1 test


----------대응 결과----------
능동대응에이전트 ID     : 120
서버측에서 받은 시간    : 20037134832
대응 레벨               : 1
대응결과 메세지         : 요청하신 작업이 성공하였습니다.
------------------------------
[root@minerva:/export/home/jsyang]
[영어][완성][2벌식]
 8 S     test  5300  5298  0  57 20        ?    201        ? 04:46:29 pts
/7    0:00 -csh
 8 S     root  5321  3781  0  41 20        ?    134        ? 04:47:09 pts
/5    0:00 grep test
[root@sirius:/export/home/jsyang/src/Responder/bin]

-----대응에이전트가 받은 메세지----------
대응에이전트 ID         : 120
받은 시간       : 200344835
대응 레벨               : 1
대응 메세지             : test
------------------------------
agent result : 0
ps -elf | grep test
 8 S     root  5325  3781  0  41 20        ?    133        ? 04:48:44 pts
/5    0:00 grep test
[root@sirius:/export/home/jsyang/src/Responder/bin]
[영어][완성][2벌식]
```

(그림 7) 사용자 세션 종료 대응 결과

(그림 7)은 "test" 계정에 대한 세션 종료 대응에 대한 시뮬레이션 결과이다. "test" 계정을 가지고 있는 사용자는 모니터링 기법에 의해 많은 자원을 소진하는 사용자임이 결정되는데, (그림 7)의 아래 터미널에는 대응 기능 동작 후, "test" 계정의 세션이 종료된 것을 볼 수 있다.

모니터링 및 대응 코드는 ANTS 기반의 실행환경을 고려하여 자바 프로그래밍 언어를 사용하였으며 다른 실행환경일 경우 각 실행환경에 해당되는 Caml이나 PLAN과 같은 액티브 네트워크 환경에 적합한 프로그래밍 언어를 사용할 수 있으며 본 논문에서 구현된 코드는 해당 언어로 포팅(porting)되어야 한다.

대응 기법 시뮬레이션은 공격으로 인한 무리한 자원의 할당에 대해 강제로 해제가 가능한지의 여부를 확인하기 위함이다.

6. 침입 감내 메커니즘

노드 생존성 보장을 위한 침입 감내 시스템의 동작 메커니즘은 모니터링 코드와 대응 코드가 유기적인 관계를 갖는다. 본 장에서는 침입 감내 시스템 구성요소의 동작 메커니즘에 대해 기술한다.

6.1 동작 메커니즘

침입 감내 시스템은 크게 3개의 구성요소로 이루어진다. 각 구성요소는 관리 도메인 내에서 동작한다.

- 관리자(Manager)

관리자는 최초에 취약성 검사를 위한 코드 배포하고, 경고 발생시 알람을 통보 받는다. 이외에 관리 대상 노드의 상태를 모니터링 한다.

- 모니터(Monitor)

모니터는 시스템 자원에 기반한 증상을 참조하여 대응을 결정하는 모니터링을 수행한다.

- 대응자(Responder)

대응자는 모니터에 의해 결정된 공격 증상에 대한 메시지를 받아 대응 기능 등을 수행한다.

〈표 2〉 웹 서버 버전 검사 코드

```
int checkWebServer()
{
    int i = 0 ;
    int sock ;
    struct in_addr addr ;
    struct sockaddr_in sin ;
    struct hostent *he ;
    unsigned long start ;
    unsigned long end ;
```

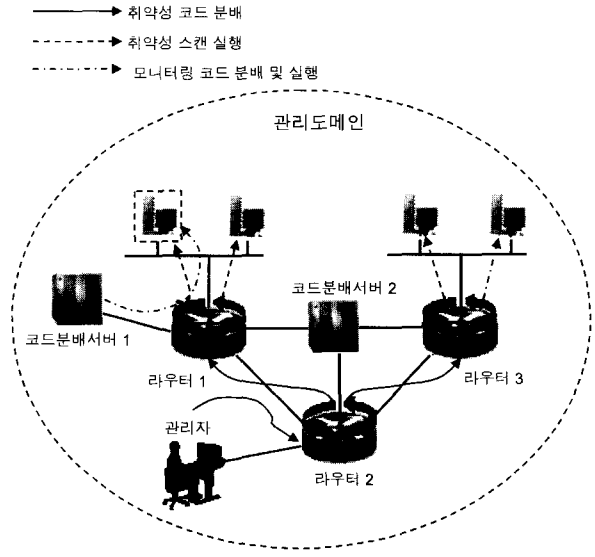
```
    char buffer[1024] ;
    char *ser_p ;
    he =gethostbyname("Target Host") ;
    start = inet_addr(he) ;
    sock = socket(AF_INET, SOCK_STREAM, 0) ;
    bcopy (he → h_addr, &sin.sin_addr, he->h_length) ;
    sin.sin_family = AF_INET ;
    sin.sin_port = htons(80) ;
    if ((connect(sock, (struct sockaddr*)&sin, sizeof(sin))) != 0)
        return (0) ;
    else
    {
        send (sock, "HEAD / HTTP/1.0\n\n", 17, 0) ;
        recv (sock, buffer, sizeof(buffer), 0) ;
        strcpy (allversion, buffer) ;
        ser_p = (strstr (buffer, "Server : ")) + 8 ;
        if (*ser_p == ' ') ser_p++ ;
        while (strchr (" \n\r", *ser_p) == 0)
        webversion [i++] = toupper (* (ser_p++)) ;
        return (1) ;
    }
    close (sock) ;
}
```

(그림 8)은 액티브 네트워크 환경에서 취약성 스캐닝 및 모니터링 코드 분배 및 실행 메커니즘을 나타낸다. 모니터링 코드가 자동적으로 배포되기 위해서는 취약성 스캔이 선행되어야 하므로 관리자는 중간 라우터(라우터 2)에 스캐닝 코드를 배포한다. 취약성 스캐닝 코드가 엣지(edge) 라우터(라우터 1, 라우터 3)에 도착하면 스캐닝을 수행한다. 취약성 스캐닝 기능은 IBAN(Intrusion Blocker based on Active Networks)에서도 언급되었던 기능으로써 취약성이 있는 노드 및 서비스를 결정하는데 중요한 기능을 한다[18].

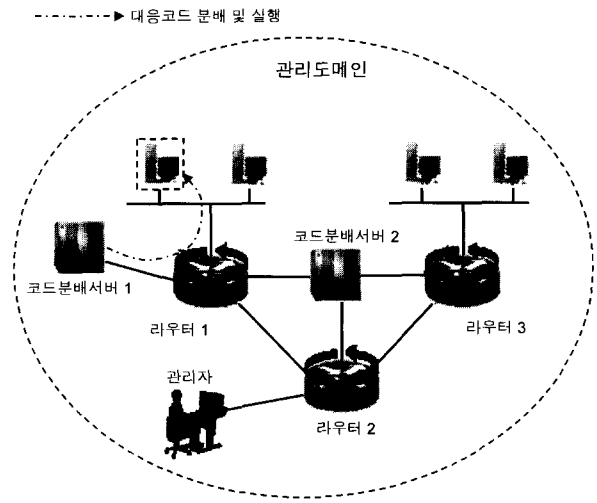
<표 2>는 웹 서버의 버전을 원격으로 검사하는 코드이다. 엣지 라우터에서 네트워크에 연결된 관리 노드에 <표 2>의 코드를 엣지 라우터에서 실행하면 목적지 노드의 웹 서버 버전을 알 수 있다. 위와 같은 방법으로 여러 가지 서비스에 대한 취약성 검사를 수행할 수 있으며 해당 노드에 알맞은 모니터링 코드를 보낼 수 있다.

모니터링 코드의 대상이 되는 시스템의 자원은 여러 가지가 존재한다. SNMP(Simple Network Management Protocol) MIB(Management Information Base)을 이용하거나 사용자 정의의 관리 정보를 정의할 수 있다[12]. 모니터링할 객체는 중간 노드와 종단 노드 특성에 따른 모니터링 객체 정의가 필요하다는 것을 고려해야한다.

취약성 스캔 후, 침입 감내 시스템은 취약성이 발견된 머신 - (그림 8), (그림 9)에서 점선이 그려진 노드 - 혹은 해당 머신이 속해 있는 네트워크의 엣지 라우터에 모니터링 코드를 배포한다.

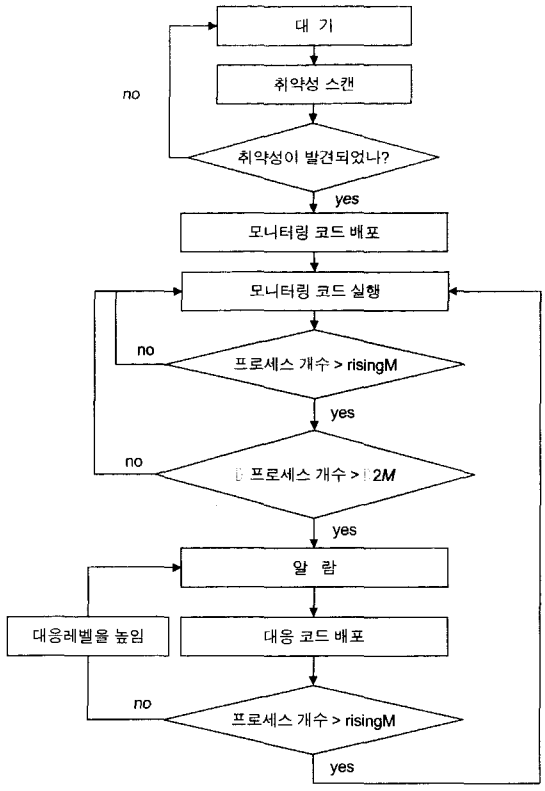


(그림 8) 액티브 네트워크 환경에서 취약성 및 모니터링 코드 분배 및 실행



(그림 9) 액티브 네트워크 환경에서 대응 코드 분배 및 실행

모니터링 코드는 취약성이 발견된 노드에서 시스템 자원을 모니터링하고 자원의 임계치가 초과되면 관리자에게 경고 알람을 발생시키고 가까운 코드분배서버1에서 해당 대응 코드를 요청한다. 대응 코드는 <표 1>에서 보는 바와 같이 대응 레벨에 따라 배포된다. 배포된 대응 코드가 임계치 이하로 프로세스의 개수를 낮추지 못할 경우 좀 더 레벨이 높은 대응 코드를 다시 요청한다. 예를 들면, 모니터링 코드가 시스템 자원을 많이 소진하는 특정 사용자를 탐지하였을 경우 대응 레벨이 "하"인 사용자 작업 지연 등의 대응을 수행할 수 있다. 그럼에도 불구하고, 임계치를 만족하지 않을 때는 대응 레벨이 "중"인 세션 종료 기능을 수행하고, 마지막 단계로 대응 레벨 "상"인 IP 블로킹 기능을 수행할 수 있다. (그림 9)는 액티브 네트워크 환경에서 대응 코드의 분배 및 실행 메커니즘을 나타낸다.



(그림 10) 침입 감내 순서도

(그림 10)은 관리자가 최초에 취약성 스캔을 시작한 후부터 대응이 종료되기까지의 순서도를 나타내었다.

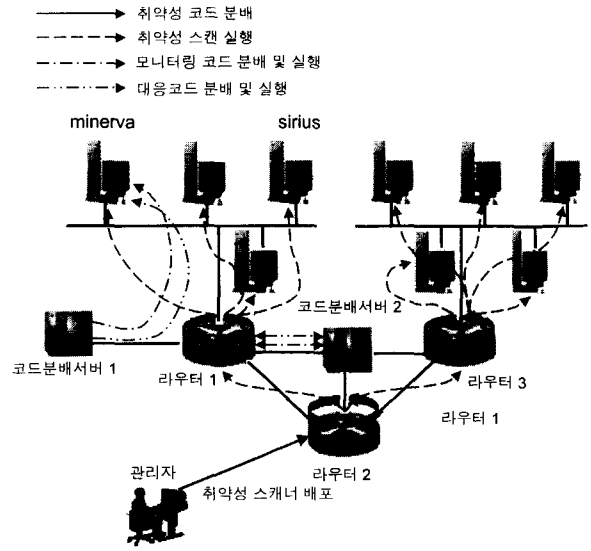
6.2 동작 시나리오

동작 시나리오는 다양하게 있을 수 있으나 중간 및 종단 노드의 예를 들어보면 다음과 같다.

● 중간 노드에서 실행 시나리오

(그림 11)과 같이 라우터2에 취약성 스캐너를 배포하면 각 네트워크의 엣지 라우터에 배포되어 해당 네트워크에 대한 취약성 스캔을 수행한다.

만약, minerva 호스트에 SQL Overflow 취약성이 발견되면 SQL 어플리케이션 모니터링 코드를 라우터1에 배포한다. 모니터링 코드는 5.1절의 모니터링 기법을 적용하여 시스템 자원이 임계치를 넘어서면 엣지 라우터에서 IP 블로킹 혹은 포트 블로킹 등의 적절한 대응 코드를 실행시킨다. 만약, 해당 대응 코드가 라우터1에 없을 경우 가까운 코드분배서버(코드분배서버2)에 요청하여 다운로드 받은 후 해당 코드를 수행한다. 대응 코드가 서버를 안전한 수준으로 만들지 못할 경우 다량의 패킷이 들어오는 인터페이스를 단절하거나 포트를 블로킹하는 등의 대응 시나리오가 가능하다. 해당 대응 코드는 취약성이 있는 서비스 혹은 어플리케이션의 패치 후 다른 노드로 이동하거나 삭제될 수 있다. 이후 관리자는 경험적인 판단에 따라 시스템의 CPU나 메모리 사용을 모니터링 하는 코드를 보내어 해당 노드의 가용성을 보장할 수 있다.



(그림 11) 액티브 네트워크 환경에서 동작 시나리오

● 종단 노드에서 실행 시나리오

어떤 네트워크에서 많은 트래픽을 유발하는 경우 해당 네트워크의 엣지 라우터에 모니터링 코드를 보내어 어떤 IP 주소가 근원지인지 찾아낸다. 그 노드가 minerva 노드이면 노드 자원을 모니터링 하는 코드를 minerva 노드에 배포하고 사용자의 자원을 모니터링 한다. 중간 노드에서 실행 시나리오와 마찬가지로 해당 노드에 코드가 없을 경우 가까운 코드분배서버(코드분배서버1)에서 다운로드 받아 모니터링 코드를 수행한다. 노드에서 많은 자원을 사용하는 사용자를 찾아낸 후 노드의 자원에 많은 무리가 있을 경우 즉시 세션을 끊어버리거나 레벨에 따라 사용자 계정 잠금이나 계정 삭제 등의 대응이 있을 수 있다.

7. 결론 및 향후 연구

본 논문은 제안된 평균 프로세스 개수에 기반한 모니터링 기법을 통해 프로세스가 점유하고 있는 자원에 대한 모니터링을 수행하고, 적절한 대응을 통해 가용성을 보장하는 메커니즘들을 기술하였으며 대응 코드는 액티브 네트워크 기반 구조에서 수행함을 가정하였다. 이는 기존의 보안 솔루션의 단점과 시스템 재설정 및 패치에 대한 수동성의 단점을 동시에 해결하는 솔루션이다. 솔루션을 적용함에 있어서 액티브 네트워크 연구에서 중요하게 고려하고 있는 성능은 시뮬레이션을 통해 평가할 필요성이 있다. 시뮬레이션 모델은 이동 코드 기술이 기반이 되므로 기존의 이동 코드 기반의 시뮬레이션 모델을 적용하여 성능 시뮬레이션을 수행할 것이다.

참 고 문 헌

[1] AN NodeOS Working Group, "NodeOS Interface Specification," Nov., 2001.

[2] AN NodeOS Working Group, "Architectural Frame-work for Active Networks," Jul., 1999.

[3] AN Security Working Group, "Security Architecture for Active Nets," Nov., 2001.

[4] Common vulnerabilities and exposures homepage, http://cve.mitre.org.

[5] D. J. Wetherall, et al., "ANTS : A Toolkit for Building and Dynamically Deploying Network Protocols," IEEE OPENARCH '98 Proceedings, San Francisco, Apr., 1998.

[6] D. L. Tennenhouse., et al., "A Survey of Active Network Research," IEEE communications magazine, pp.80-86, Jan., 1997.

[7] D. Moore, "CAIDA Analysis of Code Red," http://www.caida.org/analysis/security/code-red/.

[8] D. Sterne, et al., "Active network based DDoS defense," Proceedings of the DARPA Active Networks Conference and Exposition, May, 2002.

[9] Hi linux homepage, http://hilinux.org/.

[10] M. Cukier, et al., "Providing Intrusion Tolerance With ITUA," Conference on Dependable Systems and Networks, 2002.

[11] M. Hicks, et al., "PLAN : A Packet Language for Active Networks," ICFP, 1998.

[12] P. Grillo, S. Waldbusser, "Host Resources MIB," RFC2790, Mar., 2000.

[13] P. Pal, et al., "Survival by Defense-Enabling," Proceedings of the New Security Paradigms Workshop 2001, pp.71-78, Sept., 2001.

[14] P. Pal, et al., "Intrusion Tolerant Systems," Proceedings of the IEEE Information Survivability Workshop (ISW-2000), Oct., 2000.

[15] R. Braden, et al., "The ASP EE : An Active Network Execution Environment," Proceedings of the DARPA Active Networks Conference and Exposition, May, 2002.

[16] S. Staniford, et al., "Flash Worms : Thirty Seconds to Infect the Internet," http://www.silicondefense.com/flash/, Sept., 2001.

[17] W. A. Arbaugh, "Vulnerability Research," http://www.cs.umd.edu/~waa/vulnerability.html.

[18] W. L. Cholter, et al., "IBAN : Intrusion Blocker based on Active Networks," Proceedings of the DARPA Active Networks Conference and Exposition, May, 2002.

[19] 김현구외 5명, "액티브 노드 보안 취약성 관리 구조", KNOM Review, Vol.5, No.2, 2002.

[20] 슈퍼유저코리아 : 대한민국서버관리자그룹홈페이지, http://www.superuser.co.kr/open_lecture/solaris/page03_05.htm.

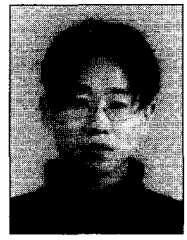
[21] 안철수 연구소 기술 기획실, "SQL_Overflow 웜의 분석 보고서", 2003.

[22] 안철수 연구소 홈페이지, http://home.ahnlab.com/securityinfo/secu_view.jsp?seq=2442.

[23] 정보홍외 2명, "침입방지시스템 기술 현황 및 전망", ITFIND 주간기술동향, 2003.

[24] 최중섭, "실시간 침입 대응 및 피해복구기술 동향", 월간 정보보호뉴스, 2000.

[25] 한국 FreeBSD 유저 그룹 홈페이지, http://www3.krfreebsd.org/handbook/.



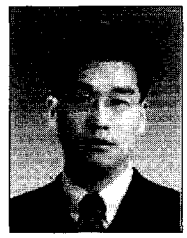
양 진 석

e-mail : jsyang@imtl.skku.ac.kr

2003년 성균관대학교 정보공학과(학사)

2003년~현재 성균관대학교 컴퓨터공학과 석사과정

관심분야 : 액티브 네트워크, 침입감내 시스템, 네트워크 관리, 네트워크 보안



이 호 재

e-mail : leehj@kisa.or.kr

1999년 성균관대학교 정보공학과(학사)

2002년 성균관대학교 전기전자 및 컴퓨터공학과(공학석사)

2002년~현재 한국정보보호진흥원 연구원

관심분야 : 컴퓨터보안, 침입감내시스템, 무선보안

장 범 환

e-mail : bchang@etri.re.kr

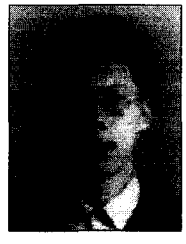
1997년 성균관대학교 전자공학과(학사)

1999년 성균관대학교 전기전자 및 컴퓨터공학과(공학석사)

1999년 성균관대학교 전기전자 및 컴퓨터공학과(공학박사)

2003년~현재 한국전자통신연구원 연구원

관심분야 : 네트워크 보안 관리, 액티브 네트워크, 네트워크 관리



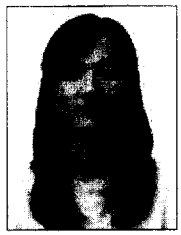
김 현 구

e-mail : hkkim@imtl.skku.ac.kr

2002년 성균관대학교 정보공학과(학사)

2002년~현재 성균관대학교 컴퓨터공학과 석사과정

관심분야 : 액티브 네트워크, 네트워크 관리, 네트워크 보안,



한 영 주

e-mail : yjhan@imtl.skku.ac.kr
1999년 성균관대학교 정보공학과(학사)
2002년~현재 성균관대학교 컴퓨터공학과 석사과정
관심분야 : 액티브 네트워크, 네트워크 보안, 침입탐지시스템

정 태 명

e-mail : tmchung@ece.skku.ac.kr
1984년 일리노이주립대학 전자계산학과 (학사)
1987년 일리노이주립대학 대학원 컴퓨터공학(공학석사)
1995년 Purdue대학교 대학원 컴퓨터공학과(공학박사)
1995년~1999년 성균관대학교 전기전자 및 컴퓨터공학부 조교수
2000년~현재 성균관대학교 전기전자 및 컴퓨터공학부 부교수
관심분야 : 액티브 네트워크, 네트워크 관리, 네트워크 보안, 통합보안 관리