알려지지 않은 악성 이동 코드에 대한 거시적 대응
(Macroscopic Treatment to Unknown Malicious Mobile Codes)

이 강 산 [†] 김 철 민 [††] 이 성 욱 [†††] 홍 만 표 [††††]
(Kangsan Lee) (Cholmin Kim) (Seonguck Lee) (Manpyo Hong)

요 약 최근 자동화된 공격기법에 의한 인프라 피해 사례가 급증하면서 효율적 대응 기법에 대한 연구가 활발히 진행되고 있다. 특히, 패킷을 통한 네트워크 서비스의 취약성을 공격하는 웜의 전파 메커니즘은 빠른 전파 속도로 인해 네트워크 대역폭 및 노드 가용성에 심각한 피해를 일으키고 있다. 이전 웜 탐지 기법들은 주로 시그너처 기반의 미시적 접근방식이 주를 이루었으나 높은 오탐지율과 조기탐지의 한계로 인해 최근에는 웜 전파의 특징에 기인한 거시적 접근 방식이 각광을 받고 있다. 본 논문에서는 패킷 마킹을 통해 웜 행동 사이클과 감염 체인으로 대표되는 웜의 행위적 특성을 탐지하고 대응할 수 있는 분산 웜 탐지 모델을 제안한다. 제안하는 웜 탐지 모델은 기존 모델들이 지닌 확장성의 한계를 분산된 호스트들의 협업적 대응으로 해결할 수있을 뿐만 아니라, 웜 탐지에 필수적인 감염 정보만을 처리함으로써 개별 호스트의 프로세싱 오버헤드를 감소시킬 수 있다. 그리고 본 논문에서 제안하는 탐지 모델 적용 시 조기 탐지 결과로 인해 웜의 감염 속도가 시간의 경과에 따라 감소되는 현상과 호스트 간의 협업적 대응에 의해 전체 호스트의 면역성이 증가되는 현상을 시뮬레이션을 통해 증명하였다.

키워드 : 웜, 컴퓨터 바이러스, 거시적 대응, 분산 웜 탐지 모델

Abstract Recently, many researches on detecting and responding worms due to the fatal infrastructural damages explosively damaged by automated attack tools, particularly worms. Network service vulnerability exploiting worms have high propagation velocity, exhaust network bandwidth and even disrupt the Internet. Previous worm researches focused on signature-based approaches however these days, approaches based on behavioral features of worms are more highlighted because of their low false positive rate and the attainability of early detection. In this paper, we propose a Distributed Worm Detection Model based on packet marking. The proposed model detects Worm Cycle and Infection Chain among which the behavior features of worms. Moreover, it supports high scalability and feasibility because of its distributed reacting mechanism and low processing overhead. We virtually implement worm propagation environment and evaluate the effectiveness of detecting and responding worm propagation.

Key words : Worm, Computer Virus, Macrospic Treatment, Distributed Worm Detection Model

1. 서 론

일반적으로 웜(Worm)은 숙주 프로그램에 기생하지 않고 독립적인 형태로 존재하며 지속적으로 자기 복제를 수행하는 악성 코드를 지칭한다. 특히 최근에는 네트워크 서버 또는 호스트의 보안상 헛점이나 정책적인 약점을 이용하여 빠르게 전파되는 유형의 웜이 주류를 이루고 있다[1,2]. 이런 방식의 웜은 사용자의 상호작용을 필요로 하지 않기 때문에 다른 유형의 악성 코드에 비해 빠른 전파 속도를 가진다. 코드레드II 웜의 경우, 불과 14시 간 만에 359,000 개의 호스트를 감염시킬 만큼 빠른 전파속도를 가지고 있었다[3].

새로운 악성 코드에 대한 분석이 이루어지기 전에 이미 최대치에 가까운 피해가 발생하는 고속전파형 웜의 등장은 시그너처 기반의 기존 악성 코드 대응 방법의 한계를 여실히 보여주었고, 네트워크 수준의 감시를 통

· 이 논문 또는 저서는 2005년 정부(교육인적자원부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임(KRF-2005-003-D00367)
† 정 회 원 : (주)콘텔라 R&D 연구소 연구원
kslee@contela.com
†† 정 회 원 : (주)시스온칩 부설연구소 연구원
cmkim@sysonchip.co.kr
††† 정 회 원 : 신구대학 인터넷정보과 전임강사
suleeip@shingu.ac.kr
(Corresponding author)
†††† 종신회원 : 아주대학교 정보및컴퓨터공학부 교수
mphong@ajou.ac.kr
논문접수 : 2006년 6월 29일
심사완료 : 2006년 9월 27일

해 이러한 유형의 새로운 악성 코드를 자동적으로 탐지하고 방어할 수 있는 시스템에 대한 연구를 촉발시켰다. 이들 대부분은 로컬 네트워크의 패킷을 감시하다가 포트 스캔 또는 동일한 페이로드를 가지는 패킷이 유의치 이상으로 증가하면 이것을 악성코드에 의한 것으로 간주하여 IP, 포트, 또는 페이로드의 특정 부분을 시그너쳐로 추출하고 해당 패킷들을 차단하는 기법들이다[4-6]. 그러나, 엄밀한 의미에서 이들 기법은 동일한 형태의 패킷을 다수 발산하는 개별 호스트의 수가 증가함을 관측할 뿐, 연쇄적인 악성 코드의 재전파 패턴을 관측의 대상으로 하는 거시적 접근의 근본 개념과는 다소 거리가 있다. 따라서 긍정 오류(False positive)를 낮추기 위해서는 유의치를 높게 설정할 수 밖에 없고 이로 인해 악성 코드의 발생 초기에 조기 탐지(Early detection)가 어렵다는 단점을 가지고 있었다.

이러한 문제점을 극복하기 위해서 최근에는 웜의 거시적 전파특성을 분석하여 알려지지 않은 새로운 웜을 조기 탐지하려는 시도가 이루어지고 있다[7-9]. 하지만 이들은 로컬 네트워크의 스위치 또는 별도의 안티바이러스 서버가 해당 네트워크의 트래픽을 분석하는 중앙집중형 기법들이다. 이러한 방식은 해당 로컬 네트워크만을 대상으로 한 분석에는 효과적으로 사용될 수 있으나, 극히 짧은 시간에 세계 곳곳으로 퍼져 나가는 최근의 웜을 조기 감지하기 위해서는 웜의 전파 경로에 있는 다수의 로컬 네트워크 감시 시스템들이 전역적인 협조체제를 이루는 것이 필수적인데, 이 때 많은 양의 정보가 다수의 감시 시스템간에 교환되고 처리 되어야 하므로 확장성(Scalability)에 문제를 안고 있다.

본 논문에서는 패킷 마킹 기법을 이용하여 네트워크에 추가적인 부담을 주지 않으면서도, 개별 호스트가 세밀한 거시적 특성을 분석함으로써 조기 탐지가 가능한 분산형 웜 탐지 시스템의 모델을 제안한다.

2. 이론적 배경

최근의 웜은 빠른 전파속도를 통해 인터넷 인프라의 대역폭을 고갈시키고 각종 서버와 호스트의 가용성을 급격히 저하시킨다. 많은 악성 프로그램 중에서도 웜의 심각성에 대한 인식이 높아지면서 최근 많은 연구가 진행되고 있다.

웜의 특성을 설명하기 위해서는 웜의 활동 사이클(Worm Cycle)과 감염 체인(Infection Chain)에 대한 정의가 필요하다. 활동 사이클이란 웜에 감염된 호스트가 다른 취약한 호스트를 감염시키기 위해서 반복하는 일련의 행위를 말한다. 웜에 감염된 호스트는 생존 기간 동안 다음과 같은 세 단계로 이루어진 활동 사이클을 반복한다.

첫 번째 단계인 잠복 단계(Dormant phase)는 웜이 활성화되기 이전 감염 호스트의 상태를 말한다. 웜은 감염된 후 즉시 활동을 시작할 수도 있지만 일정 시간의 경과나 특정한 이벤트로 인해 활성화되는 경우도 있다. 두 번째 단계는 활성화된 웜이 임의로 선택한 IP 주소를 이용하여 취약한 호스트를 탐색하는 단계(Scanning phase)이다. 이 단계에서는 임의 선택된 IP 주소와의 연결 시도를 통해 목적 호스트의 취약성을 확인한다. 마지막 단계는 탐색 단계에서 발견한 취약한 호스트에 웜의 코드가 담긴 패킷을 전달해 감염시키는 감염 단계(Infecting phase)이다[10].

웜에 감염된 호스트는 다른 호스트를 감염시키는 활동 사이클을 반복한다. 이 때 감염된 호스트는 잠복 단계를 지나 활성화된 후 또 다른 호스트를 감염시키기 위한 활동 사이클을 반복하게 되는데, 이처럼 이전의 감염 호스트가 반복한 행위를 새롭게 감염된 호스트가 반복하는 현상을 감염 체인이라고 한다. 이러한 감염 체인은 웜에 감염된 호스트의 연결 정보 및 웜 활동 사이클 사이의 인과 관계를 분석함으로써 생성 가능하다.

초기 웜 대응 연구는 급격히 증가되는 특정 패턴을 지닌 웜 패킷을 임계치 기반으로 탐지하는 방식이 주를 이루었다[4-6]. 그러나 이 같은 임계치 기반 방식의 탐지 방법은 오탐지율이 높고 조기 탐지가 어렵기 때문에 최근에는 웜의 자기 복제 특성에 기반한 탐지 연구가 진행되고 있다.

[7]에서는 웜의 근원을 탐지하기 위해 랜덤 문워크(Random moonwalk) 알고리즘을 사용하였다. 이것은 웜을 전파시킨 최초의 호스트와 전파 플로우를 식별하는 알고리즘으로써, 웜의 전파방식을 그래프 형식으로 표현하고 있으며 웜에 감염된 호스트는 vertex, 웜의 감염 경로는 edge로 표현된다. 이 알고리즘은 일정시간 동안 수집된 네트워크 트래픽으로부터 웜의 전파를 탐지하기 위해 임의의 호스트를 선택한 후, 역추적(Backward tracing)하여 의미있는 edge를 찾아낸다. 의미 있는 edge의 시작 호스트 중 가장 높은 반복 횟수를 가지는 노드가 웜 전파의 소스 호스트이다. 이 알고리즘은 각 호스트 사이의 연결관계를 알기 위해 모든 호스트의 트래픽을 감시해야 하고, 역추적의 성격 상 일정시간 동안 패킷을 저장해야 하기 때문에 실시간 탐지가 어렵다는 문제점을 지니고 있다.

한편, [8]에서는 감염된 호스트의 연결 정보를 이용하여 웜 감염 체인을 생성하는 그리디(Greedy) 알고리즘을 제안하였다. 이 알고리즘은 웜의 전파를 탐지하기 위해 일정기간 동안 수집한 네트워크 트래픽을 분석하여 호스트 간의 연결 체인을 만든다. 연결 체인의 생성은 특정 호스트간의 연결 정보에 일정시간 내에 생성된 다

른 호스트와의 연결 정보를 추가하는 방식으로 이루어진다. 이 알고리즘은 수집된 트래픽 중 호스트 간의 연결 정보에 기초한 연결 체인을 생성하며, 이러한 체인들 중에서 의심스러운 체인을 웜의 트래픽으로 간주한다.

상술된 기법들은 웜의 특성인 웜 활동 사이클과 웜 감염 체인을 탐지하기 위해 호스트의 트래픽을 수집하여 호스트 간의 연결 관계 및 패킷 정보를 분석해야 한다. 기존의 웜 탐지 시스템은 물리적인 구성 형태에 따라 다음과 같은 두가지 방식으로 분류할 수 있다[2].

첫 번째 방식은 네트워크 노드 형태의 웜 탐지 시스템으로써 로컬 네트워크의 호스트의 패킷을 모니터링할 수 있는 위치에 탐지 시스템을 설치하는 형태이다. 네트워크 노드 중심의 웜 탐지 시스템은 연결된 호스트의 패킷을 분석하여 각 호스트 간의 연결 정보를 관리함으로써 웜 사이클을 수행하는 호스트와 호스트 간의 감염 체인을 생성할 수 있다. 즉, 이 형태의 웜 탐지 시스템은 자신에게 물리적으로 연결된 모든 호스트의 트래픽을 모니터링하여 각 호스트의 연결 정보 및 패킷 정보를 분석한다. 이 방식은 내부 트래픽을 모니터링할 수 있는 위치에 패킷 모니터링을 위한 별도의 장비 또는 모듈 탑재가 필수적이다.

두 번째 방식은 네트워크의 모든 호스트에 설치된 에이전트와 별도의 탐지서버로 구성된 형태이다. 로컬 네트워크의 각 호스트에 설치된 에이전트는 다른 호스트로 전달되는 패킷을 저장한 후, 일정 시간을 주기로 웜

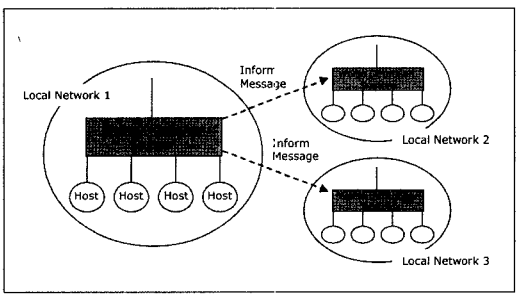

그림 1 네트워크 노드 형태의 웜 탐지 시스템

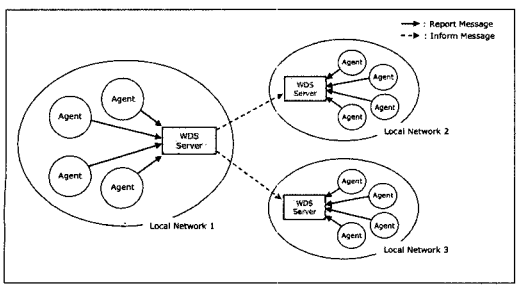

그림 2 에이전트-서버 형태의 웜 탐지 시스템

탐지시스템 서버에 보고하는 기능을 수행하며 서버는 에이전트들로부터 수집한 패킷 정보를 이용하여 웜 감염체인을 생성하는 기능을 수행한다.

이러한 중앙 집중형 웜 탐지 시스템이 가지는 가장 큰 문제는 확장성이다. 즉, 웜 탐지의 범위가 로컬 네트워크에 국한되지 않고 인터네트워크로 확장될 때에는 동일한 기능을 수행하는 탐지 시스템과의 연동 및 메세지 전달을 위한 새로운 프로토콜이 규정되어야 한다. 또한, 인터넷 수준에서의 정확한 웜 전파 그래프를 생성하고 이를 통해 웜을 감지하기 위해서는 탐지 시스템 간에 많은 양의 정보가 교환되어야 하고, 개별 탐지 시스템에서 처리해야할 정보의 양이 기하급수적으로 증가하게 된다. 특히 별도의 웜 탐지 서버가 존재하는 경우, 개별 호스트가 단순히 자신의 스캐닝 여부만을 알려주는데 그치지 않고 정확한 전파 경로 정보를 서버에 전달하려 하면, 감염을 위한 패킷과 동일한 수의 패킷이 서버에게 전달됨으로써 네트워크의 대역폭과 서버의 가용성을 급격히 저하시킬 수도 있다.

3. 분산형 웜 탐지 기법

상술된 문제의 해결을 위해 본 논문에서는 패킷 마킹을 이용하여 네트워크 트래픽 증가를 유발하지 않으며, 별도의 서버를 필요로 하지 않고 각각의 호스트가 자신에게 관련된 필수적인 감염경로 정보만을 처리하는 분산형 웜 탐지 시스템을 제안한다. 이 시스템에서 각각의 호스트에 탑재된 웜 탐지 모듈은 다음과 같은 동작을 수행한다.

- 패킷 마킹 정보 저장 : 수신된 패킷에서 특별한 마크가 발견되면 이 정보를 저장한다. 이것은 이 패킷을 보낸 호스트가 불특정 다수에 대한 스캐닝을 수행하고 있고, 이 패킷이 그 중 하나임을 의미한다.
- 스캔 감지 : 이 호스트가 다수의 불특정한 호스트에 대해 스캐닝을 하고 있는 것을 감지한다.
- 판단 및 처리 : 스캐닝 패킷을 수신한 후 일정 시간 안에 자신이 스캐닝하고 있음을 감지했다면 이에 대한 대응 작업을 수행한다. 대응 작업은 감지 전략과 파라미터에 따라 다를 수 있으며, 즉각적으로 웜이 감지되었음을 알리고 동일 형태의 패킷을 차단하거나, 좀 더 확실한 판단을 위해 다시 패킷에 마킹을 하여 보냄으로써 이를 수신한 호스트가 웜 여부를 판단하도록 할 수도 있다.

이 시스템에서 각각의 웜 탐지 모듈은 평상시에는 수신된 패킷 헤더에서 특정 필드의 마킹을 체크하고 자신이 동작 중인 호스트의 스캐닝 여부만을 검사하므로 시스템과 네트워크에 큰 부하를 주지 않는다. 따라서 기존의 상용화된 안티바이러스 소프트웨어의 일부로서 동작

하는 경량의 소프트웨어 모듈 형태로 구현이 가능하다. 또한 웜 발생 시에도 안티바이러스 개체 간의 통신에 별도의 패킷을 필요로 하지 않고, 각각의 탐지 모듈이 웜의 발생지로부터 자신까지 거쳐온 호스트 수와 같은 적은 양의 정보만으로 동작하므로 네트워크와 시스템 자원에 큰 문제를 발생시키지 않게 된다. 특히 각 호스트 간의 정보 교환만으로 웜의 거시적 특성인 감염 체인을 파악할 수 있으므로 분석 서버 또는 하드웨어 등과 같은 별도의 인프라 추가 없이 그대로 인터넷 전역을 대상으로 확장될 수 있다.

이렇듯 분산형 웜 탐지 모델에서 각 호스트는 패킷 마킹 기법을 통해 수신 호스트에게 웜의 전파 가능성을 알리는 기능을 수행한다. 패킷 마킹을 위해 이용되는 필드는 IP 헤더의 Identification 필드이다. 이 필드는 이기종 네트워크를 지나는 패킷이 MTU (Maximum trassimission unit) 크기 제한에 의해 단편화될 때 각 단편화된 패킷의 식별을 위해 이용되는 16비트 크기의 필드이다. 이 필드는 이기종 네트워크를 지나지 않을 경우 불필요한 값을 가지기 때문에 분산서비스거부 공격 시 공격의 근원을 찾는 기법들에서 이용되고 있다 [11,12].

3.1 기본 모델 (모델 1)

웜에 감염된 호스트는 동일한 취약점을 지닌 다른 호스트를 탐지하기 위해 스캐닝 또는 감염 패킷을 전송하게 되는데, 이 때 발생하는 불특정 다수에 대한 동일 패턴의 연결 시도를 카운팅함으로써 이를 탐지할 수 있다. 호스트가 웜 스캐닝으로 의심되는 행위를 탐지할 경우, 이들 패킷에 특정한 값을 마킹하여 전달함으로써 다른 호스트에게 웜의 활동 가능성을 알릴 수 있다. 마킹된 패킷을 수신한 호스트가 일정 시간 내에 이전 감염 호스트가 수행한 것과 동일한 스캐닝을 수행한다면 호스트는 이를 웜의 감염에 의한 행위로 간주하여 웜 패킷의 전송을 차단하게 된다. 그림 3은 이를 도시한 것으로 호스트 A에서 최초로 웜이 발생되었을 때 호스트 B에서 이를 감지하는 과정을 나타낸다.

최초로 웜에 감염된 호스트 A는 스캐닝을 탐지할 경우 패킷 IP 헤더의 Identification 필드에 특정 값을 마킹하여 전달함으로써 이를 수신한 호스트 B, C, F, G에게 웜의 전파 가능성을 알린다. 마킹된 패킷을 수신한 호스트인 B, C, F, G가 일정 시간 내에 호스트 A처럼 스캐닝을 탐지한다면 체인 길이가 1인 감염 체인을 관측할 수 있다. 따라서 B, C, F, G는 웜의 전파를 탐지할 수 있으며, 스캐닝에 의해 발생하는 패킷을 차단함으로써 조기에 웜의 전파를 막을 수 있다.

감염 체인을 확신하기 위해서는 연속된 스캐닝 간의 연관성을 확보해야 한다. 이를 위해서 마킹 패킷을 수신한 호스트는 마킹 패킷을 전송한 호스트와 수신 시간을 저장한다. 이 때 저장된 레코드들은 일정 시간 내에서만 유효하게 된다. 유효한 레코드의 범위를 설정하는 것은 감염 호스트 식별 및 감염 체인 생성 시 이용되는 패킷 정보의 신뢰성을 높이기 위해서이다. 마킹 패킷의 유효 시간의 범위를 TIME_START, TIME_END라고 가정한다면 마킹 패킷의 유효 시간은 웜 활동 사이클의 영향을 받는다. 웜 활동 사이클의 세 단계인 잠복 단계,

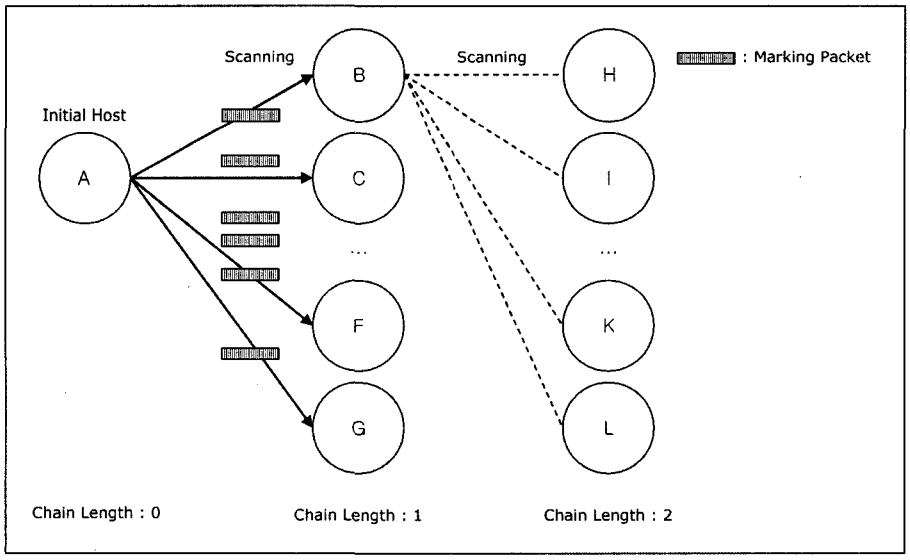

그림 3 연속적인 스캐닝 발생에 의한 감염 체인 생성 기법

스캐닝 단계, 감염 단계에서 각각 소비되는 시간을D(t), S(t), I(t)라고 가정한다면 TIME_START는 마킹 패킷의 수신 시간, TIME_END는 TIME_START + D(t)가 된다. 즉, 각각의 마킹 패킷은 TIME_START와 TIME_END 사이의 시간 동안 유효하기 때문에 마킹 패킷을 수신한 후, 해당 시간 내에 웜 활동 사이클을 탐지하지 못하면 감염 체인을 생성할 수 없으므로 웜의 전파를 탐지할 수 없다.

웜 감염 체인을 생성한 호스트는 저장하고 있는 마킹 패킷의 송신 호스트에게 패킷의 정보를 보고하여 웜 스캐닝으로 인한 패킷 전송을 차단하도록 한다. 이 보고를 통해 감염된 호스트의 수를 줄일 수 있고, 웜의 감염 속도를 감소시킬 수 있다.

3.2 마킹 카운터의 활용 (모델 2)

두 번째 모델 또한 IP 패킷의 Identification 필드에 특정 값을 마킹함으로써 수신 호스트에게 웜의 활동 가능성을 알리는 방식을 취한다. 그러나 모델 1에서처럼 연속된 스캐닝이 관측된 즉시 이것을 웜으로 판단할 때 발생할 긍정 오류를 재고하기 위하여, 마킹 카운터라고 하는 호스트 간의 연속된 연결의 수를 전달함으로써 감염 트리(Tree)의 깊이(Depth)가 일정 값 이상일 때 웜으로 간주하는 방식이다. 그림 4는 마킹 카운터를 이용한 분산 웜 탐지 시스템을 나타낸다.

최초감염 호스트인 A는 IP 헤더의 Identification 필드에 최초의 연결임을 알리는 마킹 카운터 값 1을 표시한 후, 호스트 B, C, F, G에게 전달한다. 마킹 카운터가 마킹된 패킷을 수신한 호스트인 B, C, F, G는 다른 호스트와의 연결 시, 연속된 연결이라는 알리기 위해 증가시킨 마킹 카운터를 패킷에 마킹한 후 전달하게 된다. 만약, 호스트 B에 의해 증가된 마킹 카운터 패킷을 수신한 H가 이후 M, N, O, P 등의 호스트와 연결을 시도한다면, 호스트 H는 연속된 연결의 수가 증가했다는 사실을 알리기 위해 또 다시 증가된 마킹 카운터를 마킹한 패킷을 전달한다. 즉, 마킹 카운터를 수신한 호스트가 또 다른 연결을 시도할 경우, 연속된 연결의 수를 알리기 위한 방법으로 마킹 카운터를 증가시켜 전송하는 것이다.

호스트가 수신한 마킹 카운터가 임계치 이상일 경우, 호스트는 해당 패킷을 웜의 전파로 인해 전송된 패킷으로 간주하여 필터링 한 후, 해당 패킷을 전송한 호스트에 패킷의 정보를 보고함으로써 이전 호스트가 웜 패킷의 생성을 중단하게 한다. 그러므로 호스트는 웜에 감염된 후, 분산 웜 탐지 시스템에 의해 감염 체인이 생성되고 웜 활동이 차단되는 시간 동안 그림 5와 같은 상태를 지니게 된다.

웜의 전파를 탐지한 호스트는 자신에게 마킹 카운터가 포함된 웜 코드를 전송한 호스트에 대한 정보를 유지하고 있기 때문에 해당 호스트에 웜 전파 메커니즘으로 쓰인 패킷 정보를 전달하여 호스트를 면역 상태로 변경할 수 있다. 감염 체인의 상위 단계에 속한 호스트가 웜 전파를 탐지한 호스트로부터 보고를 수신하면 해당 호스트는 또 다시 동일한 패킷을 전달한 감염 체인의 상위 노드에 연속적인 보고를 전달함으로써 감염 체인 생성에 참여한 모든 호스트가 짧은 시간 내에 면역 상태에 도달하게 된다. 이로 인해 웜의 전파가 지속되더라도 감염된 호스트의 수는 급격히 증가하지 않으며, 시간이 경과할수록 웜에 대한 전체 호스트의 면역성은 증가하게 된다.

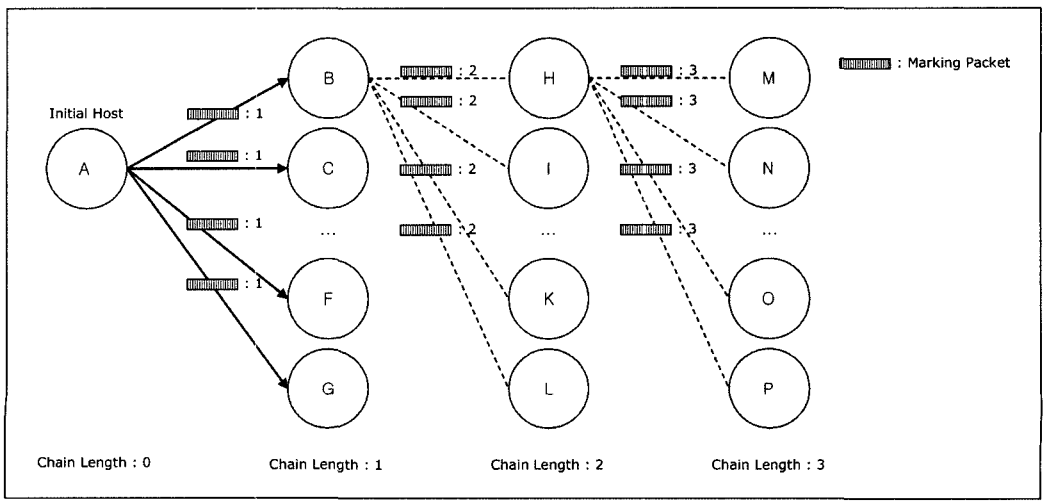

그림 4 마킹 카운터를 이용한 분산 웜 탐지시스템

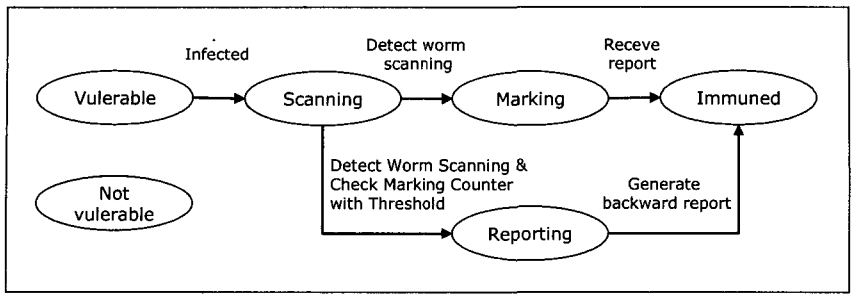

그림 5 호스트의 상태 다이어그램

이상과 같은 패킷 마킹 기반의 웜 탐지 모델을 실제로 활용하는데 있어 추가적으로 고려해야 할 점은 다음과 같다.

첫째, 정상적인 스캐닝을 웜의 활동으로 오인할 가능성이다. 일반적으로 정상 동작중인 컴퓨터가 스캐닝을 실시하는 경우는 간혹 일어날 수 있으나, 스캐닝 패킷을 받은 호스트가 일정 시간 이내에 그와 완전히 동일한 형태의 스캐닝을 실시하여 스캐닝의 체인을 형성하는 경우는 그 발생 가능성이 매우 희박하다. 하지만 정상적인 상황에서도 그러한 상황이 발생할 가능성을 아주 배제할 수는 없으므로, 스캐닝 체인의 검출 시에 해당 컴퓨터 내부의 연관 프로세스까지 확인하는 기법을 활용할 수 있다. 즉, 컴퓨터의 네트워 입출력을 관측한 결과 스캐닝의 체인이 형성되는 것으로 판단되면 즉각적으로 웜으로 간주하지 않고, 동일한 스캐닝을 수행하고 있는 주체가 스캐닝 패킷을 수신한 데몬 또는 이 데몬이 생성한 프로세스인지를 다시 확인한다면 웜의 전파 행위를 좀 더 확신할 수 있게 된다.

둘째, 웜이 패킷 마킹 값을 조작하거나, 웜 탐지 모듈의 동작을 방해하는 경우이다. 이 두 가지 문제는 결국 제안하는 기법이 적용된 안티바이러스 모듈의 안전성에 관련된 것으로 볼 수 있다. 즉, 전자는 안티바이러스 모듈이 항상 네트웍의 가장 하부 레이어에서 패킷을 감시하는 것이 보장되는가의 문제이고, 후자는 안티바이러스 모듈 자체가 항상 정상적으로 동작할 것이 보장되는가의 문제로 귀결된다. 사실 기존의 안티바이러스 제품들 또한 같은 문제를 안고 있으며, 실제로 파일 시스템 또는 네트웍 입출력에 대한 후킹(Hooking)을 통해 안티바이러스의 감지를 회피하거나 특정 안티바이러스 제품을 직접 공격하는 악성 코드들이 존재한다. 현대의 안티바이러스들은 이러한 침해 행위로부터 안티바이러스 자신을 보호하기 위한 기술을 포함하고 있으므로 제안하는 웜 탐지 기법이 현실에 적용될 때는 동안한 수준의 기술적 방어책을 적용함으로써 이러한 문제에 대응할 수 있을 것이다.

4. 실험 결과

본 논문에서 제안한 분산 웜 탐지 시스템 모델들의 웜 전파 탐지 효율을 증명하기 위한 시뮬레이션을 수행하기 위해 가상의 웜 전파 환경을 구현하였다. 가상 환경에서 시뮬레이션을 수행하여 웜 활동 사이클과 감염 체인을 생성하기 위해 각 호스트는 표 1과 같은 구성 요소를 가진 구조체로 정의된다.

표 1 호스트의 구성 요소

Property	Description
Id	호스트를 유일하게 식별할 수 있는 값
State	호스트의 상태
Dormant delay	감염된 후 스캐닝을 시작하기까지 전까지 소비하는 시간
Edge count	감염시킨 취약한 호스트의 수
Edge list	감염시킨 취약한 호스트의 리스트
Marking counter	수신한 마킹 카운터의 값

웜 전파 과정을 시뮬레이션하기 위한 알고리즘은 그림 6과 같다.

시뮬레이션은 최초의 감염 호스트를 가정한 후, 감염 호스트가 웜 활동 사이클을 수행하는 동안 감염시킨 취약한 호스트의 수를 감염 진행 단계별로 관찰하였으며, 분산 웜 탐지 시스템의 각 모델 간의 탐지 효율성을 비교하였다. 각 시뮬레이션은 표 2와 같은 파라미터를 통해 수행되었다.

그림 7은 시뮬레이션 1의 결과로써 탐지 모델을 적용하지 않은 상태에서 취약 호스트의 비율에 따른 전파 단계를 나타낸다. 실험 결과, 취약 호스트의 수가 많을수록 빠른 속도로 감염이 진행된다는 것을 확인할 수 있다. 일정 시간이 경과된 후에는 감염 호스트의 증가가 정체되는 것을 확인할 수 있는데 이것은 취약 호스트의 수가 더 이상 존재하지 않아 추가적인 감염이 이루어지지 않기 때문이다.

그림 8은 시뮬레이션 2의 결과로써 제안하는 웜 탐지 모델 1을 적용 시 취약 호스트의 비율에 따른 웜의 전

```
Procedure PropagationStage

Variables
'H' is a set of hosts in simulation environment. H[i] means ith host of H.
Each host has 4 properties
                'infected' : Boolean variable.
                'immumed' : Boolean variable.
                'propagation_count' : Integer variable, denotes the length of infection chain
                'receive_alert' : Boolean variable
'τ' is a thresh hold value to decide the flow as a worm.

Initialize
Pick certain portion of host to denote immumed ones
Pick some hosts for initially infected ones
1   while FOREVER
2       for i ← 1 to |H|
3           if H[i].infected = TRUE
4               then if H[i].immumed = TRUE
5                   then if H[i].propagation_count > τ
6                       then H[i].infected ← FALSE
7                       H[H[i].infected_from].receive_alert ← TRUE
8                       goto 2
9                   else if H[i].receive_alert = TRUE
10                      then H[i].infected ← FALSE
11                      H[H[i].infected_from].receive_alert ← TRUE
12                      goto 2
13              call WormCycle(H[i])


Function WormCycle

Input
'Hi' is an infectec host
Variables
'V' is a subset of 'H' which can be exploited by the infected host Hi.
'β' is the birth rate of the worm

1   select 'V' using Hi
2   for i ← 1 to |V|
3   Generate a real number j randomly between 0 to 1
4       if j < β
5           then V[i].infected ← TRUE
6           V[i].propagation_count ← Hi.propagation_count + 1
7   return
```

그림 6 시뮬레이션 알고리즘

표 2 시뮬레이션 파라미터

Properties	Sim. 1	Sim. 2	Sim. 3	Sim. 4	Sim. 5	Sim. 6
T_Cnt	1,000,000	1,000,000	1,000,000	1,000,000	1,000,000	1,000,000
V_Cnt/T_Cnt	30, 50, 80, 100 %	30, 50, 80, 100 %	30, 50, 80, 100 %	100%	100%	100%
A_Cnt/T_Cnt	-	100%	100%	30, 50, 80, 100 %	30, 50, 80, 100 %	100%
Scanning rate	50 hosts/sec	50 hosts/sec	50 hosts/sec	50 hosts/sec	50 hosts/sec	50 hosts/sec
Hit ratio	50%	50%	50%	50%	50%	50%
Th_Scan	-	10	10	10	10	10
Th_Length	-	1	2	1	2	1, 2, 3, 4

※T_Cnt: Total host count
※ V_Cnt: Vulnerable host count
※ A_Cnt: Agent host count
※ Th_Scan: Threshold for detecting scan
※ Th_Length: Threshold for creating an infection chain

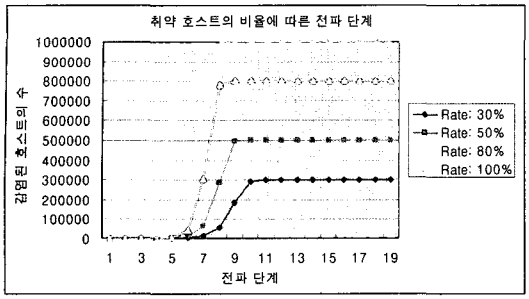

그림 7 취약한 호스트의 비율에 따른 전파 단계

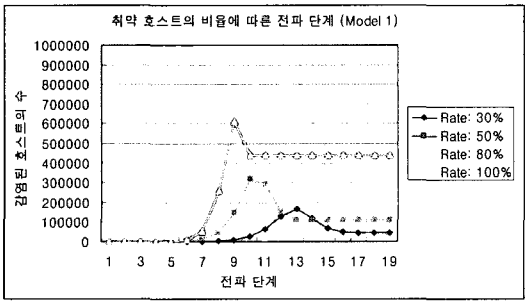

그림 8 모델 1 적용시 취약한 호스트의 비율에 따른 전파 단계

파 단계를 나타낸다. 실험 결과를 통해 취약 호스트의 비율이 높을수록 더 빠른 전파 속도를 보이지만, 일정 시간이 경과되어 웜의 전파를 탐지한 호스트의 보고에 의해 감염 체인 생성에 참여하는 호스트들이 면역 상태가 되기 때문에 급격히 감염 호스트의 수가 감소하는 현상을 확인할 수 있다. 그러나 전파 단계 후기에는 웜에 감염된 호스트가 전체 호스트 중 감염 체인을 생성할 수 있는 충분한 수의 호스트를 탐색할 수 없기 때문에 감염 상태를 벗어날 수 없으므로 일정한 수의 감염 호스트가 유지되는 현상이 나타난다.

그림 9는 시뮬레이션 3의 결과로써 제안하는 웜 탐지 모델 2을 적용 시 취약 호스트의 비율에 따른 웜의 전파 단계를 나타낸다. 실험 결과, 모델 1을 적용한 시뮬레이션 결과와 유사한 형태의 감염 호스트 증가를 확인할 수 있다. 그러나 감염 체인의 길이가 1인 모델 1의 결과에 비해 더 많은 호스트가 감염된다는 사실과 최종 감염 호스트의 수가 증가된 사실을 확인할 수 있는데, 이것은 연속적인 웜의 재전파를 탐지하기 위한 마킹 카운터의 임계치를 2로 증가시켰기 때문에 탐지 시점이 모델 1에 비해 늦어졌기 때문에 발생한 현상이다. 그리고 시간의 경과에 따른 감염 호스트 수의 변화를 관찰할 때, 마킹 카운터가 증가하더라도 웜의 전파를 탐지할 수 있다는 사실을 확인할 수 있다.

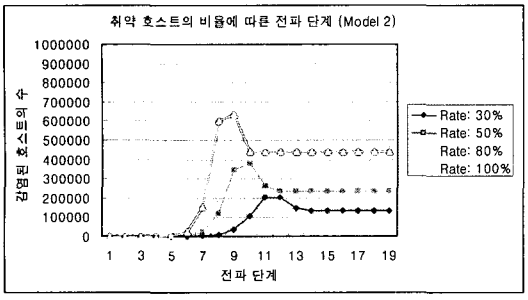

그림 9 모델 2 적용시 취약한 호스트의 비율에 따른 전파 단계

그림 10은 시뮬레이션 4의 결과로써 웜 탐지 모델 1을 적용시 에이전트의 비율에 따른 전파 단계를 나타낸다. 실험 결과를 통해 에이전트의 비율이 80% 이상일 때, 전파 탐지 후 보고에 의한 감염 호스트 감소 현상을 확인할 수 있었다. 에이전트의 비율이 80% 이하일 경우에는 웜 전파를 탐지하더라도 보고에 의한 감염 호스트 감소율이 웜 전파로 인한 감염 호스트의 증가율을 앞서지 못하기 때문에 감염 호스트의 수가 감소되는 현상이 나타나지 않았다. 즉, 웜 탐지 시스템 적용으로 인한 직접적인 효과인 감염 호스트 수의 감소는 전체 호스트 중 80%의 호스트가 웜 탐지 에이전트를 탑재하고 있어야 가능하다는 사실을 확인할 수 있다.

그림 11은 시뮬레이션 5의 결과로써 웜 탐지 모델 2를 적용시 에이전트 비율에 따른 전파 단계를 나타내는 그림이다. 실험 결과, 시뮬레이션 4의 결과처럼 에이전트의 비율이 80% 이상일 경우 웜 탐지 모델 적용으로 인한 직접적인 효과를 얻을 수 있다는 사실을 확인할 수 있다.

그림 12는 시뮬레이션 6의 결과로써 감염 체인 생성을 위한 마킹 카운터의 임계치의 변화에 따른 전파 단계와 탐지 효율성 검증을 위한 실험이다. 실험 결과를 통해 마킹 카운터의 임계치가 작을수록 감염 호스트의 증가율 또한 낮다는 사실을 확인할 수 있다. 그러나 마

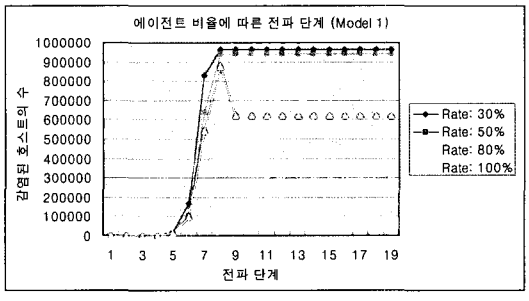

그림 10 모델 1 적용시 에이전트의 비율에 따른 전파 단계

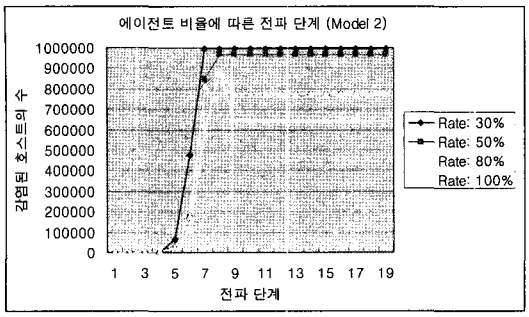

그림 11 모델 2 적용시 에이전트의 비율에 따른 전파 단계

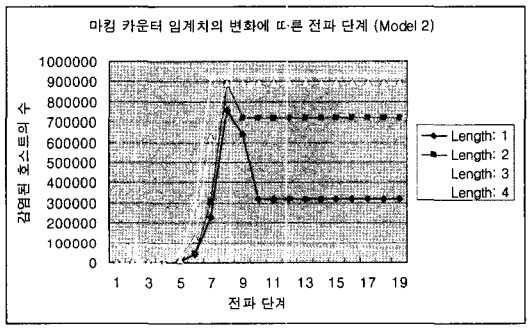

그림 12 모델 2 적용시 마킹 카운터의 임계치 변화에 따른 전파 단계

킹 카운터의 임계치가 3 이상일 경우, 감염 체인을 생성하는 속도가 웜의 전파 속도를 따라갈 수 없기 때문에 감염 호스트의 증가율이 급격히 증가한다는 사실을 확인할 수 없다. 이것은 충분한 수의 감염 체인을 생성하기 전에 전체 호스트의 감염이 완료되기 때문이다.

이상의 실험결과에서 한가지 고려할 점은 본 논문에서 제안한 모델이 웜의 탐지에 초점을 맞추고 있으므로 적극적인 대응 기법을 사용하지는 않았다는 것이다. 즉, 스캐닝의 체인을 감지한 각각의 컴퓨터는 로컬 네트웍 또는 인터넷을 통해 그 사실을 널리 전파하지 않고 오로지 자기 내부에서 실행 중인 악성 프로세스만을 종료하여 자신이 더 이상의 감염행위를 수행하지 않도록 할 뿐이다. 실험결과에서 볼 수 있듯 이러한 소극적 대응 전략을 취할 경우에는 탐지 모듈의 설치율이 어느 수준을 넘어서야 악성 코드 전파에 대한 억제 효과가 나타나게 된다.

이 같은 문제는 적극적인 대응 전략을 구사함으로서 극복할 수 있다. 한 예로서 스캐닝 체인을 감지하면 자신에게 스캐닝 패킷을 발송한 컴퓨터에게 이 사실을 통보함으로써 스캐닝 체인의 앞 단에 있는 컴퓨터도 자신을 치료하도록 하는 방법이 있다. 이 방법을 사용하면 악성 코드의 출현 초기에 감염된 컴퓨터의 수가 감소하므로 제안하는 에이전트의 설치율이 낮은 상황에서도 감염된 호스트 수가 피크에 이르는 시점을 지연시킬 수 있으며, 특히 지연된 탐지를 수행하는 두 번째 모델에서 그 효과가 더욱 크게 나타나게 될 것이다.

다른 예로는 스캐닝 체인의 발견 시 로컬 네트웍 또는 인터넷 상의 안티바이러스 서버에게 이 사실을 통보하고, 안티바이러스 서버가 관할 네트웍의 컴퓨터들에게 이 정보를 방송하는 방법을 생각할 수 있다. 이 방법은 해당 시점까지 스캐닝 패킷을 접하지 못한 컴퓨터들까지 사전에 면역 상태로 진입하는 사전 예방 효과를 발생시키게 된다.

5. 결 론

본 논문에서는 패킷 마킹을 이용하여 개별 호스트가 자신에게 필수적인 감염 경로 정보만을 분석함으로써 웜의 전파를 탐지할 수 있는 분산 웜 탐지 모델을 제안하였다. 분산 웜 탐지 에이전트가 설치된 개별 호스트들은 웜의 재전파 패턴인 감염 체인의 생성을 감시하며, 이 때 감염 정보를 공유하기 위한 방법으로 패킷 마킹 카운터를 이용한다. 마킹 카운터의 전달로 인해 각 호스트는 감염 체인에서 자신의 위치를 판단할 수 있으며 임계치 이상의 마킹 카운터를 수신할 경우, 연속적인 웜의 전파 과정을 탐지할 수 있게 된다.

한편, 이러한 웜 탐지 기법을 현실적으로 적용하기 위해서는 다음과 같은 문제를 고려하여야 한다.

첫째는 정상적인 스캐닝을 웜의 활동으로 오인하는 긍정 오류를 최소화 하는 기법이 병행 활용되어야 한다는 것이다. 정상적인 동작 중에 스캐닝의 체인이 형성되는 경우는 극히 드문 것이 사실이나 호스트 내의 프로세스 흐름까지 고려하는 등의 방법으로 탐지 오류율을 최소화할 수 있는 기법이 적용되는 것이 바람직하다.

둘째는 제안하는 웜 탐지 기법이 적용된 에이전트의 설치율이 낮은 상황에서도 효과를 발휘할 수 있도록 적극적인 대응 방법이 적용되어야 한다는 것이다. 이를 위해서는 감염 체인의 상위 호스트에게 웜의 전파 메커니즘으로 이용된 패킷 정보를 연속적으로 보고하거나, 감염 체인에 속하지 않은 인접 호스트에 통지하는 등, 보다 많은 호스트들에게 면역성을 제공하는 적극적 대응 모델을 개발하고 그 효과를 분석하여야 한다.

이 같은 문제는 향후 연구를 통해 해결할 것이다.

참 고 자 료

[1] Nicholas Weaver, Vern Paxson, Stuart Staniford and Robert Cunningham, A taxonomy of computer worms, In WORM'03 October 27, 2003, ACM,

Whshington, DC, USA.
[2] Stuart E. Schechter, Jaeyeon Jung and Arthur W. Berger, Fast Detection of Scanning Worm Infections, Recent Advances in Intrusion Detection, 7th International Symposium, RAID 2004, Sophia Antipolis, France, September 15-17, 2004.
[3] Shigang Chen and Yong Tang, Slowing down Internet worms, In Proceedings of The 24th International Conference on Distributed Computing System (ICDCS'04), March 2004, IEEE, Tokyo, Japan.
[4] Kim, H.A. and Karp, B., Autograph: Toward Automated, Distributed Worm Signature Detection, in the Proceedings of the 13th Usenix Security Symposium (Security 2004), San Diego, CA, August, 2004.
[5] Sumeet Singh, Cristian Estan, George Varghese and Stefan Savage, Automated Worm Fingerprinting, Proceedings of the ACM/USENIX Symposium on Operating System Design and Implementation, San Francisco, CA, December 2004.
[6] Nicholas Weaver, Stuart Staniford and Vern Paxson, Very Fast Containment of Scanning Worms, Proceedings of the 13th Usenix Security Conference, 2004.
[7] Yinglian Xie, Sekar, V., Maltz, D. A., Reiter, M. K. and Hui Zhang, Worm Origin Identification Using Random Moonwalks, Security and Privacy, 2005, IEEE Symposium.
[8] Al-Hammadi, Y. and Leckie, C., Anomaly Detection for Internet Worms, Integrated Network Management, 2005, IM 2005, 2005 9th IFIP/IEEE International Symposium.
[9] Cholmin Kim, Seong-uck Lee andManpyo Hong, Macroscopic Treatment to Polymorphic E-mail Based Viruses, LNCS 3045(Springer-Verlag), Computational Science and Its Applications - ICCSA 2004, pp.867-876, 2004, 5.
[10] Zesheng Chen, Lixin Gao and Kevin Kwiat, Modeling the spread of active worms, In IEEE INFOCOM 2003, IEEE, April 2003.
[11] Heeran Lim and Manpyo Hong, Effective Packet Marking Approach to Defend against DDoS Attack, Computational Science and Its Applications, ICCSA 2004, International Conference, Assisi, Italy, May 14-17, 2004.
[12] A. Perrig, D. song and A. Yaar, Pi: A Path Identification Mechanism to Defend against DDoS Attacks, In Proceedings of the 2003 Security and Privacy Symposium, May 2003.

이 강 산

2004년 8월 아주대학교 정보 및 컴퓨터공학부 졸업. 2006년 8월 아주대학교 정보통신전문대학원 정보통신공학과 석사 졸업. 2006년 8월~현재 (주) 콘텔라 R&D 연구소 연구원

김 철 민

2000년 아주대학교 정보 및 컴퓨터공학부 졸업(학사). 2002년 아주대학교 정보통신 전문대학원 정보통신공학과(석사). 2004년 아주대학교 정보통신 전문대학원 정보통신공학과 수료(박사). 2004년~현재 시스온칩 부설 연구소 연구원

이 성 욱

1994년 2월 아주대학교 공과대학 컴퓨터공학과 학사. 1996년 2월 아주대학교 대학원 교통공학과 석사. 2003년 2월 아주대학교 대학원 컴퓨터공학과 박사. 2003년 3월~현재 신구대학 인터넷정보과 전임강사

홍 만 표

1981년 서울대학교 계산통계학 전공(학사). 1983년 서울대학교 계산통계학 전공(석사). 1991년 서울대학교 병렬처리 전공(박사). 1983년~1985년 울산공과대학 전임강사. 1993년~1994년 미네소타 대학 교환 교수. 2000년~2001년 조지워싱턴 대학 교환 교수. 1985년~현재 아주대학교 교수