情報保護學會論文誌
第17卷 第5號, 2007. 10

에이전트 기반의 공격 정보 수집 시스템 설계 및 구현*

김 익 수,† 김 명 호‡
숭실대학교

Design and Implementation of an Agent-Based System for Luring Hackers

Ik-su Kim,† Myung-ho Kim‡
Soongsil University

요 약

허니팟은 해커에게 의도적으로 공격을 허용하는 컴퓨터 자원으로서 해커에 대한 공격 정보를 수집하는데 목적이 있다. 그러나 인터넷에 존재하는 무수히 많은 컴퓨터들 중에서 해커가 허니팟을 공격할 가능성은 매우 적다. 이를 보완하고자 사용하지 않는 포트를 기반으로 해커의 공격 정보를 수집하는 시스템들이 개발되었지만, 실제 환경에 적용하기에는 많은 문제점이 있다. 본 논문에서는 기존 시스템들의 문제점을 보완한 에이전트 기반의 공격 정보 수집 시스템을 제안한다. 이 시스템은 해커를 유인하여 공격 정보를 수집하며, 해커의 공격으로부터 클라이언트들을 보호한다. 또한, 제안 시스템은 공격 정보를 수집하기 위해 클라이언트의 자원을 사용하기 때문에 추가적인 IP 주소와 하드웨어의 낭비를 줄일 수 있다.

ABSTRACT

A honeypot is a security resource whose value lies in being attack. It collects data regarding the attack strategies and tools of hackers. However, the honeypot is normally located at a single point, and the possibility is small that a hacker will attack it. Unused ports-based decoy systems which gather data about hackers activities have been developed to complement honeypots. However, the systems have some problems to be deployed in actual environment. In this paper, we propose an agent-based system which enhances shortcomings of the unused ports-based decoy systems. It makes honeypot gather more information regarding hacker activities and protects clients from attacks. Moreover, the proposed system can increase the chance of tracking hackers activities without wasting additional IP addresses and computer hardwares.

Keywords : *Computer security, Agent, Signature, Honeypot, Intrusion*

Ⅰ. 서 론

최근 컴퓨터와 인터넷의 발달로 멀티미디어를 포함하는 다양한 형태의 정보들을 쉽고 편리하게 이용할 수 있게 되었으며, 쇼핑, 은행, 주식거래와 같은 업무도 인터넷을 통해 신속하게 처리할 수 있게 되었다. 그러나 불행히도 해커들은 이러한 기술들을 악용하여 개인의 프라이버시 침해는 물론, 기업 및 공공기관의 서비스를 마비시킴으로써 많은 피해를 주고 있다. 게다가 바이러스와 인터넷 웜에 의한 많은 피해는 컴퓨터와

접수일: 2007년 4월 12일; 채택일: 2007년 8월 6일
* 본 연구는 숭실대학교 교내연구비 지원으로 이루어졌습니다.
† 주저자, skycolor@ss.ssu.ac.kr
‡ 교신저자, kmh@ssu.ac.kr

인터넷 보안의 중요성을 절실히 느끼게 하고 있다.

지금까지 해커에 의한 다양한 형태의 공격들을 탐지하고 차단하기 위해 안티바이러스, 침입탐지시스템, 침입차단시스템을 포함한 많은 보안 시스템들이 개발되어 왔다[1,2]. 이 시스템들은 대부분 알려진 공격 유형에 관한 정보를 기반으로 공격에 대응하기 때문에 해커의 공격 전략과 공격 도구들에 관한 정보 수집은 보안 시스템들의 능력을 좌우하는 중요 요소라 할 수 있다.

허니팟은 해커에게 의도적으로 공격을 허용하는 컴퓨터 자원으로서 해커에 대한 공격 정보를 수집하는데 목적이 있다[3-6]. 허니팟에 의해 수집된 정보는 새로운 공격을 탐지하고 차단하기 위해 여러 보안 시스템들에 의해 사용된다. 그러나 해커가 인터넷에 존재하는 무수히 많은 컴퓨터들 중에서 허니팟을 공격할 가능성은 매우 적기 때문에 해커에 대한 많은 공격 정보를 수집하는데 한계가 있다. 허니팟을 통해 해커들의 공격 정보를 충분히 얻기 위해 단순히 여러 네트워크에 많은 허니팟을 배치할 수 있지만, 이는 추가적인 하드웨어 및 IP 주소의 낭비가 따르며, 설치된 많은 허니팟 관리 및 수집된 정보를 분석하기 위한 인적 자원의 낭비를 초래할 수 있다. 그리고 사용하지 않는 포트를 이용하여 해커를 유인하고 공격 정보를 수집하는 허니트랩[7]과 유인포트 시스템[8]이 개발되어 왔지만 이 시스템들은 해커에게 쉽게 노출될 수 있으며, 실제 환경에 적용하기에는 여러 문제점이 있다.

본 논문에서는 사용하지 않는 포트를 이용하여 해커를 유인하는 기존 시스템들의 문제점을 보완한 에이전트 기반의 공격 정보 수집 시스템을 제안한다. 제안 시스템은 공격 정보를 수집하기 위해 사용되는 IP 주소와 하드웨어 낭비를 줄이기 위해 인터넷 상의 가용한 클라이언트 자원을 이용한다. 기존 시스템과 달리 제안 시스템은 해커에게 노출되는 것을 막기 위해 중요 데이터를 변조하며, 해커의 공격에 의해 발생할 수 있는 시스템 및 네트워크 과부하를 예방한다. 또한, 시스템 보안 기능을 통해 해커의 공격으로부터 클라이언트를 보호하며, 보안 프로그램의 시그너처 파일을 주기적으로 업데이트함으로써 새로운 공격에 민첩하게 대응한다. 제안 시스템은 클라이언트의 운영환경에 맞게 자동 설정되기 때문에 실제 환경에서 매우 유연하게 동작한다.

Ⅱ. 관련 연구

해커는 인터넷 상의 특정 컴퓨터를 공격하기 전에 OS 핑거 프린팅과 포트 스캐닝을 통해 해당 컴퓨터의 정보를 수집한다. 수집된 정보에는 설치된 운영체제의 종류와 버전, 현재 가용한 서비스 유형과 관련된 취약점 정보가 포함된다. 이들 정보를 기반으로 해커는 목표를 공격하여 쉘을 획득하기도 하며, 악성 프로그램을 설치함으로써 여러 사이트의 아이디와 패스워드를 획득하기도 한다. 게다가 자동화 된 공격 도구인 인터넷 웜을 통해서 단기간에 많은 컴퓨터들을 공격하기도 한다. 인터넷 웜은 몇몇의 패킷 전송을 통해 쉽게 보안에 취약한 컴퓨터를 찾아 공격한다. 이러한 공격에 효율적으로 대응하기 위해서는 악의적인 트래픽과 시스템 내부에서 발생하는 해커들의 활동에 관한 정보를 신속하게 수집 및 분석하는 것이 매우 중요하다. 이 장에서는 해커에 대한 공격 행위를 식별하고 이에 관련된 정보를 수집하기 위해 개발된 연구 기술들에 대해 설명한다.

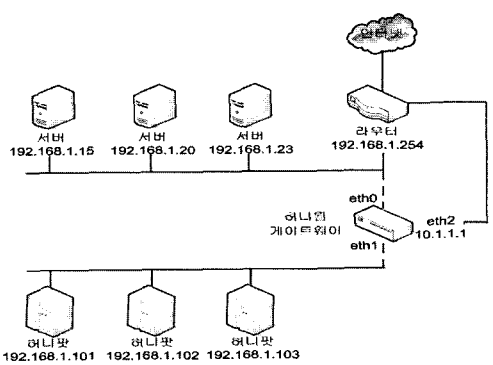
2.1 RTSD(Real Time Scan Detector)

RTSD는 국내 정보통신망에 대한 스캔 공격을 탐지하고 적극 대응하고자 한국 정보보호 진흥원에서 개발한 실시간 스캔 공격 탐지 도구로서 네트워크 취약점 검색공격을 탐지하는 실시간 검색탐지 시스템(Real Time Scan Detector Agent)과 이러한 탐지시스템으로부터의 결과 값을 분석하고 실시간으로 공격에 대응하는 실시간 검색탐지 관리시스템(Real Time Scan Detector Manager)으로 구성된다[9]. 에이전트의 스캔 공격 탐지 방법은 한 호스트에서 일정 시간 간격으로 일정한 수의 연결 요청이 있을 경우 취약점 검색 공격이라 간주한다. 에이전트 프로그램은 하나의 네트워크 또는 서브 네트워크의 여러 사용자 컴퓨터에 설치되어 실행되며 네트워크에서 발생하는 스캔 공격을 탐지하여 로그를 남기며, 매니저 프로그램에게 그 결과를 통보한다. 매니저 프로그램은 전체 네트워크를 감시하며, 탐지된 공격에 자동 대응하는 기능을 갖는다. 그러나 RTSD는 알고리즘의 단순성으로 인해 탐지 기능이 스캔 공격에 국한된다는 단점이 있다.

2.2 허니넷

보안 관련 연구기관에서는 새로운 공격에 대한 안티 바이러스와 침입탐지시스템의 시그너처를 생성하기 위해 허니넷을 사용하고 있다. 허니넷은 컴퓨터와 인터넷 보안을 위해 고의로 해커로부터의 공격을 허용함으로써, 알려지지 않은 해커의 공격 전략과 공격 도구에 대한 자료를 수집하기 위해 하나 이상의 허니팟으로 구성된 네트워크다[10,11]. 허니팟은 일반 사용자들에게 서비스를 제공하기 위한 자원이 아니라 해커를 유인하기 위한 자원이기 때문에 허니팟에 접근하는 모든 행위는 해커들의 불법 행위로 간주된다. 따라서 허니팟에서 발생하는 모든 행위와 허니넷에서 발생하는 모든 트래픽은 분석 가치가 상당히 높다.

[그림 1]은 일반적인 허니넷 구축 모델을 나타낸다. 해커는 외부로부터 허니팟을 공격할 수 있으며 공격에 성공한 해커는 또 다른 허니팟을 공격할 수 있다. 그러나 허니팟으로부터 네트워크에 존재하는 일반 서버나 외부 인터넷에 존재하는 컴퓨터로의 공격은 허니월(Honeywall)에 의해 차단된다[12]. 허니넷이 많은 해커들을 유인하기 위해서는 해커들에게 쉽게 발견되어야 한다. 그러나 인터넷 상에는 무수히 많은 컴퓨터들이 존재하기 때문에, 해커가 허니넷에 존재하는 허니팟을 공격할 가능성이 적다. 해커가 허니팟을 공격할 가능성을 높이기 위해 각각의 허니팟들은 여러 네트워크에 독립적인 서버로 구축되거나, VMWare 혹은 Virtual PC와 같은 가상화 소프트웨어를 통해 구축될 수 있다. 각 가상 컴퓨터는 각각의 IP 주소를 가지고 있으며 다른 가상 컴퓨터에 다른 운영체제를 동작시킬 수 있다. 하지만 운영체제에 따라 가상 서버를 위한 라이선스 비용을 지불해야 하는 단점이 있다.



[그림 1] 허니넷 구축 모델

현재 국내에서는 KrCERT(인터넷침해사고대응지원센터)가 매월 허니넷을 통해 웜, 바이러스 현황을 파악하고 있으며, 브라질 역시 2003년 9월 NBSO/Brazilian CERT와 정보통신부산하 소프트웨어 평가기관인 CenPRA가 공동으로 브라질 허니팟 연합을 결성하고 분산 허니팟 프로젝트를 시작하여 현재 약 20여개 기관이 가입하여 운영 중이다[13,14].

2.3 허니트랩

허니트랩은 TCP 서비스에 대한 공격을 관찰하기 위한 네트워크 보안 프로그램이다[7]. 기존의 허니팟이 사용하지 않는 IP 주소로의 접근을 비정상적인 행위로 간주하는 것과 달리 허니트랩은 사용하지 않는 포트로의 접근을 비정상적인 행위로 간주한다. 허니트랩은 특정 포트로 유입되는 패킷을 다른 호스트나 다른 서비스로 리다이렉트하는 것은 물론, 모든 트래픽을 기록하는 기능도 제공한다. 따라서 허니트랩이 많은 컴퓨터에 설치되고 비정상적인 행위로 간주되는 모든 연결들을 허니팟으로 리다이렉트할 경우, 해커의 공격 행위에 관한 정보들을 효율적으로 수집할 수 있다. 그러나 허니트랩은 해커들에 의해 존재가 쉽게 노출될 수 있다는 문제가 있다. 즉, 허니트랩에 의해 리다이렉트된 해커가 허니팟에서 ifconfig나 netstat 명령을 수행할 경우, 이 프로그램들은 허니트랩이 설치된 컴퓨터의 정보가 아닌 허니팟 정보를 제공하기 때문에 허니트랩과 허니팟의 존재가 노출될 수 있다. 게다가 허니트랩은 포트의 전송 속도를 제어할 수 없기 때문에 허니트랩이 설치된 컴퓨터는 해커의 악의적인 행위에 의해 커다란 부하를 가져올 수 있다. 물론 허니트랩의 아이디어는 충분히 가치가 있지만 클라이언트들은 허니트랩을 설치하려 하지 않는다. 이는 허니트랩이 해커의 정보 수집을 위한 목적을 가진 보안 관련 연구자에게는 유용하지만 클라이언트에게는 적합한 서비스를 제공하지 않기 때문이다.

2.4 유인포트 시스템

유인포트 시스템[8]은 해커를 유인하기 위해서 서버에서 현재 사용되고 있지 않은 포트를 유인포트로 사

용한다. 유인포트는 허니트랩에서 공격 정보 수집을 위해 사용되는 포트와 기능이 유사하지만, 이 포트로 인해 발생하는 시스템 및 네트워크 부하에 따라 전송 속도를 제어할 수 있다는 장점이 있다. 유인포트 시스템은 해커의 공격 정보를 수집하기 위한 유인포트들의 최대 수신 패킷량을 조절하기 위해 한계값을 설정하고, 한계값에 도달할 경우에 패킷 전송의 지연 시간을 높여 수신되는 패킷량을 한계값 이하로 유지한다. 또한, 서버 보안을 위해 이 포트에 접근했던 해커가 서비스 포트에 접근할 경우 허니팟으로 리다이렉트하기 때문에 해커로부터 서비스 프로그램을 보호할 수 있다. 하지만 허니트랩과 마찬가지로 유인포트 시스템은 해커에 의해 쉽게 노출될 수 있으며, 서버를 기반으로 하여 동작하기 때문에 유인포트를 생성하는데 많은 제한이 있다. 즉, 서버가 사용 중인 서비스 포트를 유인포트로 사용할 수 없기 때문에 서비스 프로그램을 공격하는 해커의 정보를 수집하는데 한계가 있다. 또한, 다수의 서버에 설치되는 유인포트 시스템을 운영환경에 따라 효율적으로 관리하기 위한 방법이 제시되지 않았다.

요약하면, 허니넷은 미리 예약된 IP 주소에 접근하는 해커들의 공격 정보만을 수집하기 때문에 정보 수집에 많은 제한이 있다. 또한 사용되지 않는 포트를 이용하여 공격 정보를 수집하는 허니트랩과 유인포트 시스템은 실제 환경에서 운용되기에 여러 문제점이 있다. 이를 보완하기 위해 본 논문에서 제안하는 시스템은 유인포트의 기능을 개선한 디코이 포트(Decoy Port)를 이용하여 해커의 공격 정보를 수집한다. 디코이 포트는 사용되지 않는 포트를 이용하여 생성된다는 점에서 유인포트와 유사하지만, 서버 상에서 생성되는 유인포트와 달리 클라이언트 상에서 생성되기 때문에 다양한 포트번호를 가질 수 있는 장점이 있다.

Ⅲ. 시스템 요구 사항 및 구조

3.1 시스템 요구 사항

앞서 살펴본바와 같이 공격 정보를 수집하기 위한 기존 시스템들이 효율적으로 운영되기 위해서는 개선되어야할 여러 문제점들을 가지고 있다. 이러한 문제점을 개선하고 실제 환경에서 유연하게 동작하는 시스템 구현을 위해서 제안 시스템은 다음과 같은 요구 사항이 만족되어야 한다.

첫째, 공격 정보를 수집하기 위해서 가용한 클라이언트의 자원을 이용해야 한다. 유인포트 시스템은 서버의 자원을 이용하기 때문에 서비스 질을 낮출 수 있다. 또한 서버 프로그램에 의해 자주 사용되는 포트들은 유인포트로 사용될 가능성이 적기 때문에 해당 포트로의 공격 정보를 수집하기가 어렵다. 하지만 클라이언트는 서버 프로그램에 의해 자주 사용되는 포트 번호를 거의 사용하지 않기 때문에 다양한 포트들을 이용하여 공격 정보를 수집할 수 있다.

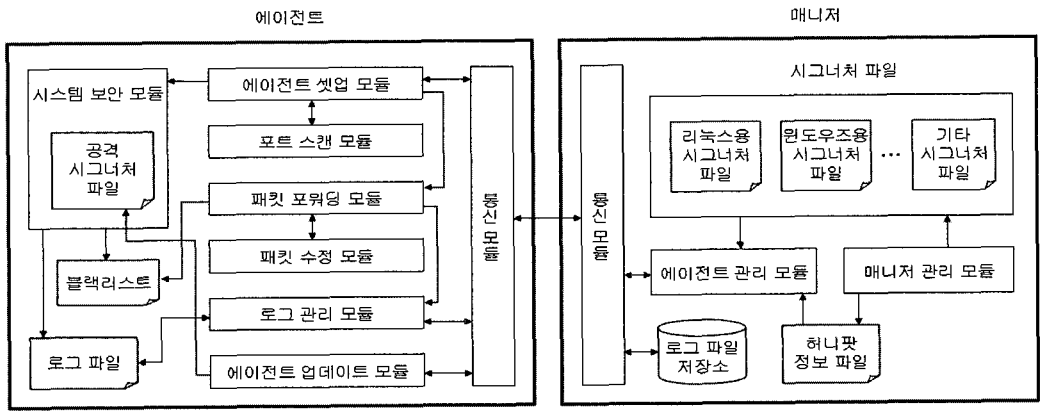
둘째, 제안 시스템은 디코이 포트에 의해 포워딩되는 데이터를 필요에 따라 변조해야 한다. 해커가 클라이언트 상의 디코이 포트에 의해 허니팟으로 리다이렉트 되면, 해커에 의한 모든 명령 결과는 허니팟으로부터 생성된다. 이 때 생성된 결과에 허니팟 IP 주소 정보가 포함될 경우, 해커는 자신이 공격한 클라이언트 IP 주소와 다르다는 것을 알 수 있다. 따라서 해커로부터 이러한 사실을 숨기기 위한 데이터 변조 기능이 필요하다.

셋째, 클라이언트의 제안 시스템 설치를 유도하기 위해서는 클라이언트에게 보안 서비스를 제공해야 한다. 허니트랩은 공격 정보 수집을 위한 목적을 가진 보안 숙련자에게 유용하지만, 클라이언트에게는 불필요한 도구일 뿐이다. 해커의 공격으로부터 클라이언트를 보호하는 보안 기능을 제공함으로써 클라이언트들이 제안 시스템을 설치하도록 유도한다.

넷째, 제안 시스템은 클라이언트 환경과 현재 운영중에 있는 허니넷 환경에 가장 적합한 시스템 설정이 이루어져야 한다. 이를 위해서는 에이전트와 매니저 구조가 제안 시스템에 적합하다. 에이전트는 클라이언트 환경에 맞는 시스템 설정 기능을 제공하고, 매니저는 원활한 에이전트 설정을 위한 다양한 정보를 제공해야 한다.

3.2 시스템 구조

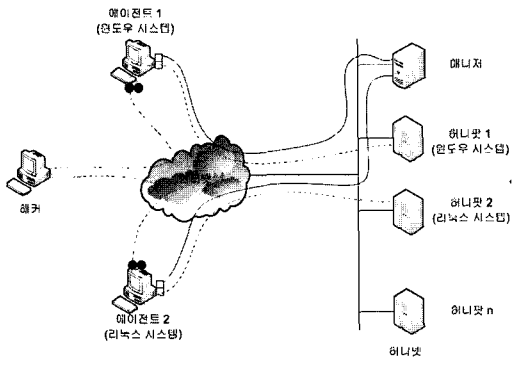
제안 시스템은 에이전트와 매니저 프로그램으로 구성된다. 에이전트는 인터넷상의 가용한 클라이언트에 설치되며, 공격 정보를 수집하기 위해 해커를 유인하고 해커로부터 클라이언트를 보호한다. 매니저는 별도의 서버에 설치되며, 에이전트 프로그램이 실행될 때의 초



〔그림 2〕 제안 시스템의 전체 구조도

기화 설정과 주기적인 에이전트 프로그램의 업그레이드를 담당한다. [그림 2]는 제안 시스템의 전체 구조도를 나타낸다. 에이전트는 8개의 모듈과 공격 시그너처 파일, 로그 파일, 블랙리스트로 구성된다. 매니저는 3개의 모듈과 에이전트의 시그너처 파일을 갱신하기 위해 사용되는 최신 버전의 시그너처 파일들, 허니넷에 배치되는 허니팟의 정보를 저장하기 위한 파일, 에이전트로부터 수신된 로그 파일을 보관하는 로그 파일 저장소로 구성된다.

[그림 3]과 같이 에이전트는 해커를 유인하기 위해 클라이언트에서 현재 사용되고 있지 않은 포트를 디코이 포트로 사용한다. 이 포트는 실제로 서비스를 위해 열린 포트가 아니기 때문에 이 포트로의 접근은 비정상적인 행위로 판단된다. 따라서 이 포트로 유입되는 패킷들은 공격 정보 수집을 위한 목적으로 사용된다. 에이전트는 클라이언트의 운영체제 종류에 따라서 윈도우용 에이전트는 윈도우 시스템에, 리눅스용 에이전트는 리눅스 시스템에 설치된다. 매니저는 디코이 포트가 패킷을 포워딩해야 할 목적지 IP 주소를 결정하는 역할을 하며, 이 IP 주소는 에이전트가 설치된 클라이언트의 운영체제에 따라 상이하다. 즉, 윈도우용 에이전트는 포워딩하는 목적지가 윈도우용 허니팟 IP 주소로 설정되며, 리눅스용 에이전트는 리눅스용 허니팟 IP 주소로 설정된다. 따라서 해커가 윈도우 시스템의 디코이 포트에 접근할 경우에는 윈도우용 허니팟으로 리다이렉트되며, 리눅스 시스템의 디코이 포트에 접근할 경우에는 리눅스용 허니팟으로 리다이렉트 된다. 매니저는 에이전트가 새로운 공격에 민첩하게 대응할 수 있도록 주기적으로 최신의 공격 시그너처를 제공한다.



〔그림 3〕 에이전트와 매니저의 배치도, 점선 : 해커의 공격에 의해 생성되는 패킷 흐름, 실선 : 에이전트와 매니저간의 패킷 흐름

Ⅳ. 시스템 설계

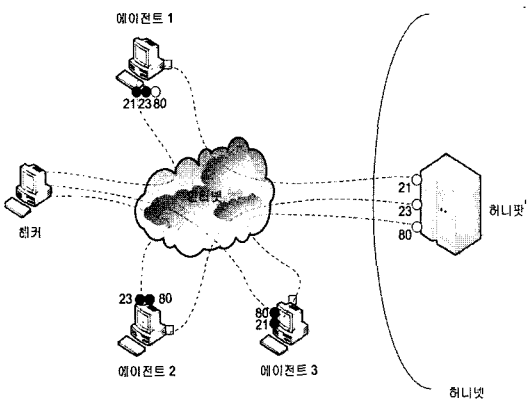
4.1 에이전트 셋업 모듈

에이전트 셋업 모듈은 에이전트를 초기화하는 모듈로서 매니저에게 초기화 설정 요구를 보낸다. 해커의 공격을 허니넷에 존재하는 허니팟으로 리다이렉트하기 위해서는 클라이언트의 운영체제와 일치하는 허니팟의 IP 주소를 알아야 한다. 즉 윈도우 운영체제가 설치된 클라이언트를 공격하는 해커는 윈도우용 허니팟으로 리다이렉트 되어야 하며, 리눅스 운영체제가 설치된 클라이언트를 공격하는 해커는 리눅스용 허니팟으로 리다이렉트 되어야 한다. 이를 위해 에이전트 셋업 모듈은 현재 클라이언트의 운영체제 정보를 매니저에게 전

달하며, 매니저로부터 동일한 운영체제가 설치된 허니팟의 IP 주소를 수신한다. 또한 허니팟이 해커의 공격을 허용하기 위해 열어둔 포트 번호를 매니저로부터 수신함으로써 에이전트는 디코이 포트를 생성할 수 있다. 만일 에이전트가 무작위로 디코이 포트를 생성할 경우 디코이 포트를 관리하는 프로세스로 인한 컴퓨팅 자원을 낭비할 수 있으며, 과중한 네트워크 트래픽을 유발할 수 있다. 따라서 현재 허니팟이 처리할 수 있는 서비스에 해당하는 포트 번호를 전달하여 불필요하게 디코이 포트가 생성되는 것을 막을 수 있다. 에이전트 셋업 모듈은 수신된 허니팟 IP 주소와 포트 번호를 이용하여 디코이 포트 프로세스를 생성한다. 즉, 디코이 포트 프로세스는 현재 클라이언트에서 사용되고 있지 않은 포트들중에서 매니저로부터 수신된 포트번호와 일치하는 포트를 디코이 포트로 사용하며, 이 포트에 접근하는 해커들을 허니팟으로 리다이렉트 한다.

클라이언트가 서버 프로그램의 목적으로 특정 포트를 사용할 경우 해당 포트가 이미 디코이 포트로 사용되고 있을 가능성이 있다. 이 경우에는 사용 중인 포트를 닫아야 하는데 단순히 에이전트를 종료한 후 에이전트 셋업 모듈을 재기동하여 새로운 환경 설정을 적용시킬 수 있다. 또한, 에이전트 셋업 모듈을 재기동하지 않고 포트를 닫기 위해서는 해당 포트의 관련 프로세스를 종료함으로써 포트를 닫을 수 있다.

[그림 4]는 에이전트에 의해 생성된 디코이 포트가 해커의 공격을 허니팟으로 리다이렉트하는 모습을 나타낸다. 에이전트 1이 설치된 컴퓨터가 현재 개인 홈페이지 서비스를 위해 80번 포트를, 허니팟은 해커의 공격을 허용하기 위해 21, 23, 80번 포트를 열었다고 가



〔그림 4〕 디코이 포트의 선택과 허니팟으로의 리다이렉션

정할 때, 에이전트는 매니저로부터 허니팟이 21, 23, 80번 포트를 열어두고 있다는 정보를 수신하기 때문에 21, 23, 80번 포트를 디코이 포트로 사용할 것이다. 그러나 에이전트는 미사용 포트만을 이용하여 디코이 포트를 생성할 수 있기 때문에 에이전트는 21, 23번 포트만을 디코이 포트로 사용한다. 마지막으로 에이전트 셋업 모듈은 운영체제 종류에 따른 최신 시그너처를 매니저로부터 수신하고 보안 프로그램을 실행한다.

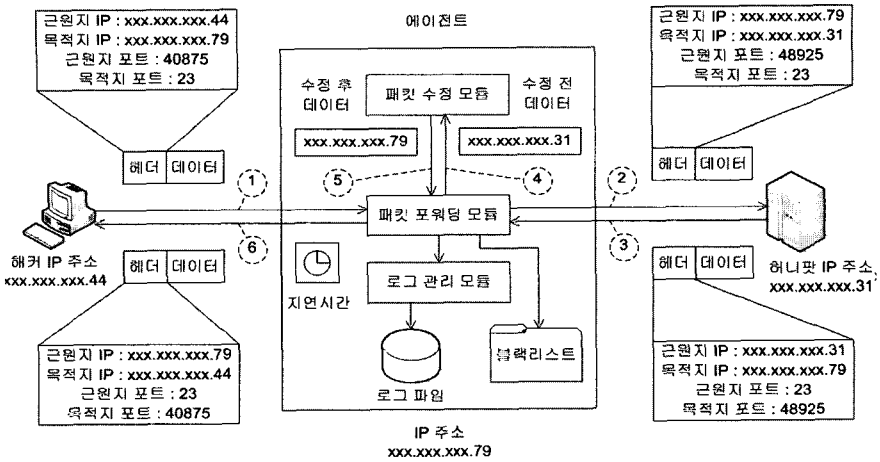
4.2 포트 스캔 모듈

포트 번호는 잘 알려진 포트, 등록된 포트, 동적 포트로 나뉜다. 잘 알려진 포트는 0번부터 1023번까지 사용되며, 대부분의 시스템에서 시스템 프로세스 또는 권한이 있는 사용자가 실행하는 프로그램이 사용한다. 한편 등록된 포트는 1024번부터 49151번까지 사용되며 일반 사용자가 실행하는 프로그램이 사용한다. 동적 포트는 49152번부터 65535번까지 사용되며 운영체제에 의해서 자동으로 클라이언트 프로그램에 할당되는 포트들이다.

해커들은 시스템의 취약 서비스 프로그램을 공격하기 위해 잘 알려진 포트와 등록된 포트를 공격한다. 따라서 포트 스캔 모듈은 디코이 포트로 사용할 수 있는 포트 번호를 찾기 위해 0번부터 49151번 포트를 스캔한다. 윈도우나 리눅스는 netstat 명령을 제공하기 때문에 포트 스캔 모듈은 이 명령을 통해 현재 사용 중인 포트 목록을 쉽게 알 수 있다. 사용 중인 포트 목록은 에이전트 셋업 모듈에 전달되며, 에이전트 셋업 모듈은 매니저로부터 수신된 포트 목록 중에서 현재 클라이언트의 미사용 포트 번호와 일치하는 포트를 디코이 포트로 사용한다.

4.3 패킷 포워딩 모듈

디코이 포트로 유입되는 패킷은 비정상적인 패킷들이기 때문에 허니팟으로 리다이렉트되어 분석될 필요가 있다. 에이전트는 리다이렉트 모드 혹은 대기 모드로 동작하는데, 리다이렉트 모드로 동작하는 에이전트의 패킷 포워딩 모듈은 이러한 비정상적인 패킷들을 허니팟으로 포워딩하는 역할을 한다. 반면, 대기 모드로 동작하는 에이전트의 패킷 포워딩 모듈은 비정상적



〔그림 5〕 패킷 포워딩 모듈에 의한 재전송과 패킷 수정 모듈에 의한 데이터 변조

인 패킷을 로그 관리 모듈로 전달한다.

해커에 의해 생성된 패킷이 클라이언트의 디코이 포트로 유입되면 [그림 5]와 같은 과정에 따라 처리된다.

① 패킷의 헤더가 제거된 후 패킷 데이터가 패킷 포워딩 모듈에게 전달된다.
② 허니팟으로 패킷 데이터를 재전송한다. 이 때, 패킷 헤더의 목적지 주소는 허니팟의 IP 주소로 변경된다.
③ 허니팟에 의해 처리된 결과가 패킷 포워딩 모듈로 유입된다.
④ 패킷 포워딩 모듈이 패킷 수정 모듈에게 데이터를 전달한다.
⑤ 패킷 수정 모듈에서 수정된 데이터가 패킷 포워딩 모듈에게 전달된다.
⑥ 해커에게 패킷 데이터를 재전송한다. 이 때, 패킷 헤더의 목적지 주소는 해커의 IP 주소로 변경된다.

이 때, 해커가 허니팟으로 DOS 공격을 할 경우 클라이언트는 디코이 포트의 패킷 포워딩으로 인한 과도한 부하를 피할 수 없다. 이러한 공격으로부터 클라이언트를 보호하기 위해 패킷 포워딩 모듈은 수신 버퍼의 크기를 제한하며, 일정시간의 지연 후 허니팟으로 재전송한다. 지연 시간은 해커가 콘솔 프로그램에서 키보드 입력을 할 때 끊김 현상 없이 즉각적으로 반응하기에 충분히 작은 시간으로 설정되며, 초당 데이터 전송량은 텍스트나 이진파일을 전송하기에 충분하게 설정된다.

디코이 포트를 통해 유입된 패킷은 공격에 의한 패킷이기 때문에 패킷 포워딩 모듈은 해당 패킷의 근원지 IP 주소를 블랙리스트에 기록하며, 해당 IP 주소를 가지는 해커로부터 클라이언트를 보호하기 위한 보안 정책을 위해서 시스템 보안 모듈에게 블랙리스트가 갱신되었다는 사실을 통보한다.

4.4 패킷 수정 모듈

리다이렉트된 해커가 허니팟으로의 침입에 성공한 후 ifconfig와 같은 시스템 명령을 수행하면 허니팟의 IP 주소를 확인할 수 있다. 이는 해커 자신이 공격 대상으로부터 다른 컴퓨터로 리다이렉트되는 사실을 알려주는 것과 같다. 따라서 허니팟과 에이전트의 존재를 숨기기 위해서는 패킷 데이터에 포함된 허니팟의 IP 주소를 클라이언트 IP 주소로 변조하는 기능이 필요한데 패킷 수정 모듈이 이러한 기능을 담당한다. [그림 5]에서 알 수 있듯이 수신된 패킷의 데이터에서 허니팟의 IP 주소가 포함된 문자열을 발견하면 클라이언트의 IP 주소 문자열로 대체하여 리다이렉트 사실을 숨긴다.

4.5 에이전트 업데이트 모듈

제안 시스템은 에이전트를 기반으로 해커를 유인하기 때문에 인터넷상에 존재하는 많은 컴퓨터에 설치되어야만 많은 공격 정보를 수집할 수 있다. 이를 위해서

에이전트는 보안 프로그램을 제공함으로써 많은 클라이언트들이 에이전트를 설치하도록 유도한다. 에이전트는 나날이 등장하는 새로운 공격에 대응하기 위해 보안 프로그램에 사용되는 시그너처 파일을 지속적으로 업데이트해야 한다. 에이전트 업데이트 모듈은 이러한 업데이트 기능을 하는 모듈로서 주기적으로 매니저에게 최신의 시그너처 파일을 요청한다. 이러한 시그너처 파일의 요청은 에이전트가 초기화 될 때 반드시 수행되며, 운영 중에도 최신의 시그너처 파일을 유지하기 위해 주기적인 업데이트 요청을 매니저에게 전달한다. 에이전트 업데이트 모듈은 현재 에이전트에 포함된 시그너처 파일 버전을 매니저에게 전송하며, 매니저는 수신된 파일의 버전과 가용한 최신 시그너처 파일의 버전을 비교하여 업데이트가 필요할 경우 에이전트에게 새로운 시그너처 파일을 전송한다. 따라서 에이전트는 새롭게 등장하는 해커의 공격이나 인터넷 웜 공격에 민첩하게 대응할 수 있다.

4.6 로그 관리 모듈

대기 모드로 동작하는 에이전트의 로그 관리 모듈은 디코이 포트에 접근하는 비정상적인 행위를 기록한다. 이 모드로 동작하는 에이전트는 인터넷 웜과 같이 자동화된 공격에 대한 정보를 수집하는데 적합하다. 에이전트에 포함된 보안 프로그램은 매니저에 의해 제공된 시그너처를 기반으로 공격을 탐지한다. 만일 디코이 포트를 통해 유입된 패킷들이 보안 프로그램에 의해 탐지되지 않을 경우에는 새로운 인터넷 웜 공격일 가능성이 높기 때문에 로그 파일에 기록된다. 디코이 포트로 유입되는 모든 패킷을 허니팟으로 포워딩할 경우에는 클라이언트를 비롯하여 매니저 서버와 허니팟에게 커다란 부하를 유발시킬 수 있기 때문에 로그 관리 모듈은 단지 보안 프로그램에 의해 탐지되지 않은 로그만을 매니저로 전송한다. 전송된 로그파일은 새로운 공격 시그너처를 생성하는데 사용된다. 특히, SSH와 같이 데이터가 암호화 되는 경우에는 해당 포트가 대기 모드로만 동작된다. 즉, 리다이렉트 모드와는 달리 해커로부터 유입되는 패킷을 수신만 하며, 패킷 데이터가 제외된 헤더 정보만이 수집된다. 이는 해커의 공격을 탐지하기 위해 데이터 매칭이 아닌 헤더 매칭을 위한 시그너처 생성을 가능하게 한다.

4.7 시스템 보안 모듈

에이전트가 설치된 클라이언트를 보호하기 위해서는 해커의 불법적인 행위를 식별하고 차단하는 기능이 필요하다. 앞서 기술했듯이 제안 시스템의 디코이 포트는 서비스를 제공하는 포트가 아니기 때문에 이 포트로의 접근은 해커의 공격으로 간주되어 해커의 IP 주소 정보가 블랙리스트에 기록된다. 또한, 시그너처를 기반으로 해커의 공격을 탐지하고 차단하기 위해 시스템 보안 모듈은 네트워크 기반의 침입 탐지를 수행하며, 패턴 매칭을 통해 해커의 침입을 식별하면 시스템 보안 모듈은 패킷의 근원지 IP 주소 정보를 블랙리스트에 기록한다. 블랙리스트에 등록된 해커들이 에이전트가 설치된 클라이언트에 접근할 때, 이를 차단하기 위해 시스템 보안 모듈은 IPTables을 이용한다. 패킷 포워딩 모듈로부터 블랙리스트의 변경 통보를 받을 경우 다음과 같은 IPTables 룰을 생성한다.

iptables -A INPUT -p TCP -s Hacker_IP --dport Normal_Ports -j DROP

Hacker_IP는 IPTables에 의해 차단될 해커의 IP 주소를 나타내며, Normal_Ports는 디코이 포트를 제외한 현재 열려있는 포트번호를 나타낸다. DROP을 통해 지정된 Hacker_IP 주소로부터 Normal_Ports에 유입되는 패킷을 무시하고 버리게 된다. 물론, 처음 에이전트가 실행될 때에는 에이전트 셋업 모듈이 블랙리스트에 존재하는 모든 IP 주소를 사용하여 위에 명시된 IPTables 룰을 생성한다.

에이전트의 설치로 인해 발생할 수 있는 추가적인 보안 취약점에 대응하기 위해 이 모듈은 프로세스를 감시하여 악의적인 프로세스의 생성을 막아야 한다. 일반적으로 해커들은 열린 포트와 관련된 서비스 프로그램의 취약점을 수집하여 취약점이 노출된 서비스 프로그램에 쉘 코드 공격을 수행한다. 쉘 코드는 쉘을 실행시키는 코드로서, 버퍼 오버플로우나 포맷스트링 공격은 이러한 쉘 코드를 취약 프로그램이 사용 중인 스택에 삽입하여 쉘 코드를 실행시킨다. 따라서 해커가 에이전트의 취약점을 공격하여 쉘을 획득하게 되면 루트 권한의 쉘을 획득하는 결과를 가져온다.

[그림 6]은 에이전트의 취약점을 공격하여 쉘을 획

```
static asmlinkage int sec_sys_execve()
{
    ......
    filename=getname((char *) regs.ebx);
    if(strstr(filename, "/bin/sh")) &&
     !(strcmp(current->session, psid)))
        return 0;
    ......
}
```

〔그림 6〕 에이전트 취약점 공격에 의한 쉘 획득을 막는 커널 모듈 코드

득하려는 시도를 막기 위한 커널 모듈 코드이다. 커널이 시스템 명령을 처리하기 위해서는 execve 시스템 콜 함수를 호출하게 되는데, 원본 execve 시스템 콜 함수를 [그림 6]에 명시된 코드로 대체함으로써 불법적인 쉘 획득을 막을 수 있다. [그림 6]의 filename은 현재 수행되는 시스템 명령에 대한 문자열이 저장된 변수이며, current->session은 현재 프로세스의 세션 ID이다. 여기에서 current는 커널 내의 현재 프로세스 상태를 나타내는 task_struct 구조체의 포인터 변수를 나타낸다. psid 는 에이전트의 세션 ID를 나타내며, 에이전트가 실행될 때 커널 모듈이 수신하게 된다. 커널 모듈 코드는 해커의 공격에 의한 쉘 프로그램 실행을 막기 위해서, 현재 실행되는 명령이 쉘 프로그램이고 현재 실행중인 프로세스의 세션 ID 가 에이전트의 세션 ID와 동일하면 이를 해커의 공격으로 간주하여 쉘 프로세스 생성을 중지한다. 만일, 쉘 명령뿐만 아니라 다른 명령들을 등록할 경우 해당 명령들의 실행도 차단할 수 있다.

4.8 매니저 관리 모듈

에이전트가 디코이 포트를 생성하고 이 포트에 접근하는 해커들을 허니팟으로 리다이렉트하기 위해서는 허니팟이 서비스하는 포트 번호와 IP 주소를 알아야 한다. 또한 에이전트가 새로운 공격에 빠르게 대응하기 위해서는 새로운 시그너처가 지속적으로 제공되어야 한다. 매니저는 윈도우와 리눅스를 포함한 여러 운영체제에 대한 최신의 시그너처와 허니넷을 구성하는 허니팟에 대한 정보를 제공한다. 허니팟 정보 파일은 허니넷을 구성하는 허니팟들의 IP 주소와 서비스 포트 번호, 설치된 운영체제 정보를 포함한다. 매니저 관리 모듈은 허니팟 정보 파일과 시그너처 파일을 편집 및 등록하기 위한 GUI를 제공함으로써 매니저 프로그램의 관리를 용이하게 한다. 따라서 관리자는 새로운 허니팟이 허니넷에 추가되거나, 허니넷의 IP 주소 및 서비스 포트 번호를 수정할 경우 GUI를 통해 쉽게 재설정할 수 있으며, 운영체제 종류에 따라 새로운 공격 시그너처 파일을 쉽게 등록할 수 있다.

4.9 에이전트 관리 모듈

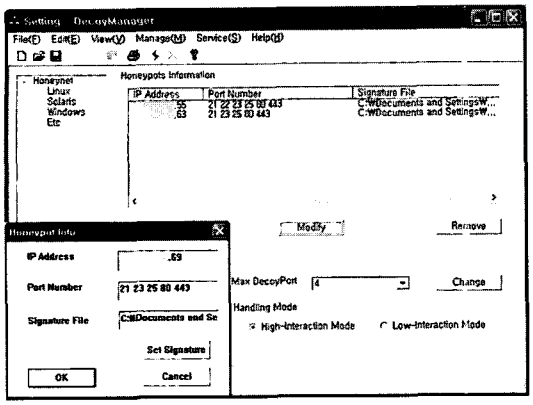
에이전트 관리 모듈은 에이전트 프로그램의 운영환경에 따른 에이전트의 초기화 및 업데이트를 담당한다. 에이전트 관리 모듈은 에이전트의 초기화 요청을 수신할 때 해당 에이전트가 설치된 클라이언트의 운영체제 종류를 인지하여 동일한 운영체제가 설치된 허니팟의 IP 주소와 서비스 포트를 전달함으로써 에이전트가 디코이 포트를 생성할 수 있게 한다. 에이전트의 초기화 요청 메시지에는 시그너처의 업데이트를 위한 버전 정보가 포함되어 있으며 에이전트 관리 모듈은 수신된 버전 정보와 매니저가 보유한 최근의 시그너처 파일의 버전을 비교한다. 만일 최근 버전이 아닐 경우, 이 모듈은 에이전트에게 최근의 시그너처 파일을 전송하여 에이전트가 새로운 공격에 대응할 수 있게 한다.

4.10 통신 모듈

통신 모듈은 에이전트와 매니저간의 네트워크 통신을 담당하는 모듈이다. 에이전트 셋업 모듈에 의한 초기화 및 에이전트 업데이트 모듈의 업데이트 작업에 필요한 메시지 전송과 파일 전송 작업을 하며, 주기적인 로그 관리 모듈에 의한 로그 파일 전송 작업을 수행한다.

V. 구현 결과 및 실험

[그림 7]은 허니넷 관리를 위한 GUI 기반의 매니저 프로그램을 나타내며, 매니저 관리자는 트리 뷰를 통해 다수의 허니팟을 운영체제 종류에 따라 관리할 수 있다. 트리 뷰에서 운영체제 종류를 클릭하면 해당 운영체제가 설치된 허니팟 목록을 볼 수 있고, 허니넷에 새로운 허니팟을 추가하거나 기존 허니팟을 삭제 및 수정하기 위한 인터페이스도 함께 제공한다. 새로운 허니

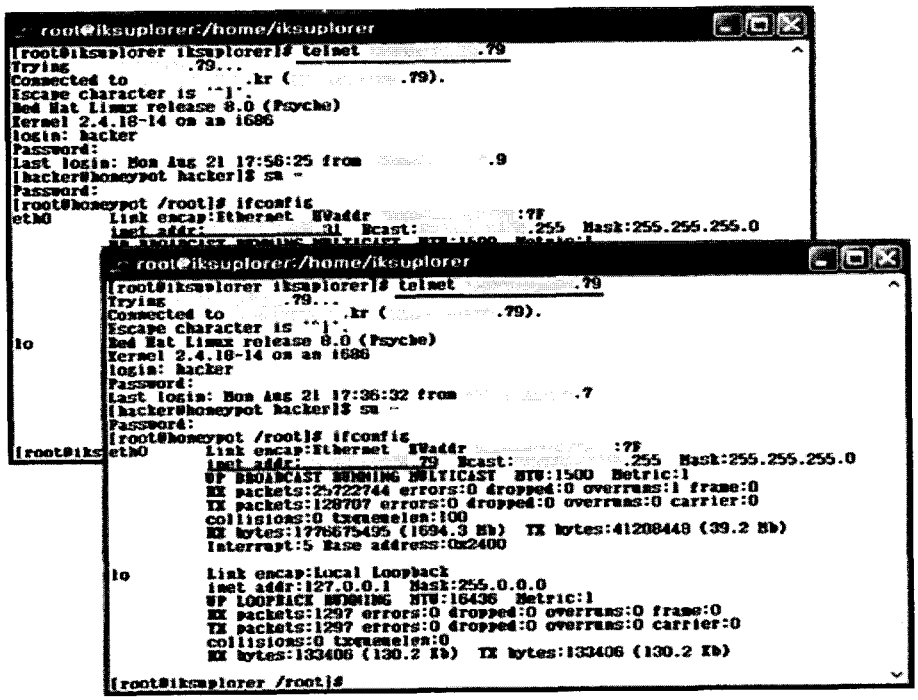


〔그림 7〕 허니팟 정보 관리를 위한 매니저 인터페이스

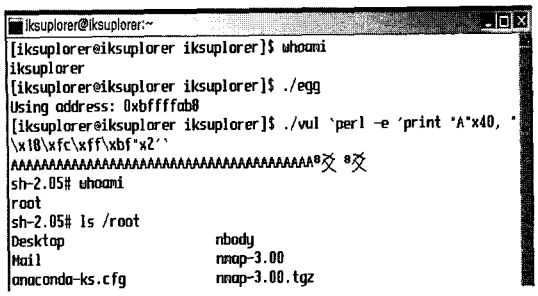
팟을 추가하기 위해서는 허니팟에 할당되는 IP 주소와 해커의 공격을 허용하기 위한 포트번호, 에이전트에 의해 사용될 시그너처 파일을 지정한다.

기존의 허니트랩이나 유인포트 시스템에서는 해커가 공격에 성공한 후 ifconfig 명령을 수행하면 허니팟의 IP 주소가 확인되기 때문에 리다이렉트되고 있다는 사실을 알 수 있다. 제안 시스템은 이를 숨기기 위해서 데이터 변조 기능을 제공한다. 데이터 변조 기능이 없을 경우에는 호스트 ID가 79인 클라이언트를 공격했을 때 ifconfig 명령을 통해 확인되는 호스트 ID가 [그림 8]의 좌측 결과와 같이 허니팟의 호스트 ID인 31이다. 하지만 에이전트는 패킷 수정 모듈을 제공하여 우측 결과와 같이 허니팟의 호스트 ID를 클라이언트의 호스트 ID로 변조하기 때문에 해커는 리다이렉트되고 있다는 사실을 인식하지 못한다.

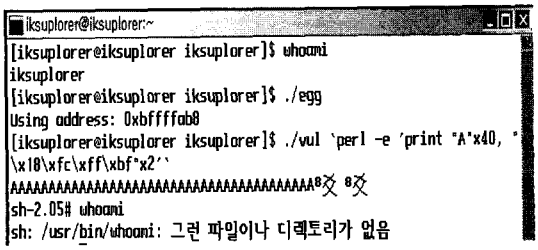
해커가 우연히 에이전트의 존재를 식별하고 취약점을 이용한 쉘 코드 공격을 수행할 경우에는 시스템 보안 모듈이 이를 차단한다. 시스템 보안 모듈의 보안 기능 실험을 위해서 실제로 에이전트에 대한 리모트 공격이 이루어져야 하지만 에이전트는 버퍼 오버플로우나 포맷 스트링 취약점을 최소화하기 위해 입력 길이를 제한하는 보안 프로그래밍 기법을 이용하여 구현되었기 때문에 에이전트의 취약점을 찾아 쉘 코드 공격을 하기에는 어려움이 많다. 이러한 관계로 쉘 코드 공격에 취약한 프로그램을 작성하고 로컬에서 공격을 수행하였으며, 이 때 시스템 보안 모듈이 쉘 코드 공격을 차단하는지의 여부를 알아보았다. [그림 9]는 시스템 보안 모듈이 설치되지 않은 상태에서 쉘 코드 공격이 이루어졌을 때를 나타내며, [그림 10]은 시스템 보안 모듈이 설치된 후 해커의 쉘 코드 공격이 차단되는 모



〔그림 8〕 에이전트의 패킷 수정 모듈에 의한 허니팟 주소 변조



〔그림 9〕 시스템 보안 모듈이 없을 때의 쉘 코드 공격 결과



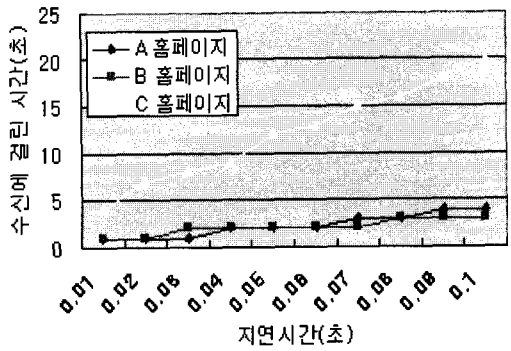
〔그림 10〕 시스템 보안 모듈이 존재할 때의 쉘 코드 공격 결과

습을 나타낸다. [그림 9]에서 볼 수 있듯이 시스템 보안 모듈이 설치되지 않은 상태에서는 루트 쉘을 획득한 후에 아무런 제약 없이 모든 명령을 수행할 수 있지만, [그림 10]에서와 같이 시스템 보안 모듈이 설치된 후에는 시스템 명령을 수행할 수 없는 것을 알 수 있다.

에이전트는 클라이언트의 자원을 이용하기 때문에 디코이 포트를 통해 전달되는 패킷량을 조절하여 클라이언트의 시스템 및 네트워크 부하를 최소화해야 한다. 그러나 포워딩되는 패킷량이 너무 적을 경우에, 오히려 해커가 현재 공격중인 대상을 포기하고 더 빠르게 응답하는 다른 공격 대상을 찾을 것이다. 현재 컴퓨터 및 네트워크 상황을 고려하면, 실시간 요청에 따른 응답 시간이 3초 이상 걸릴 경우 사용자들은 서비스 질에 큰 불만을 갖는다. 마찬가지로 해커도 지연시간이 긴 대상을 공격하기보다 오히려 공격이 원활히 이루어질 수 있도록 빠른 응답 시간을 제공하는 대상을 공격할 가능성이 높다. 따라서 해커가 대상을 공격하기에 만족스러우며 클라이언트의 자원을 보호할 수 있는 수준의 패킷 포워딩 모듈의 적정 지연시간을 설정해야 한다. 이를 위해 실험에서는 웹 브라우저를 이용하여 여러 홈페이지에 접속하고, 패킷 포워딩 지연 시간에 따른 홈페이지별 초기화면의 수신 완료 시간을 측정하였다. 실험은 패킷 포워딩 모듈의 전송 지연 시간을 0.01초

〔표 1〕 홈페이지별로 디코이 포트에 의해 포워딩되는 패킷량

실험 대상 홈페이지	포워딩 되는 패킷량(KB)
A 홈페이지	38
B 홈페이지	28
C 홈페이지	168



〔그림 11〕 지연 시간에 따른 홈페이지 콘텐츠 수신 완료 시간

부터 0.1초까지, 데이터 전송크기를 1024바이트로 설정하여 수행되었다. 전송 지연 시간의 최대값을 0.1초로 제한한 이유는 키보드 입력에 반응하는 시간이 0.1초 이상이 될 경우 콘솔 프로그램에서 끊김 현상이 컸기 때문이다.

[표 1]은 브라우저를 통해 실험 대상 홈페이지에 접속했을 때 디코이 포트에 의해 포워딩되는 패킷량을 나타내며, [그림 11]은 에이전트의 패킷 포워딩 지연 시간에 따라 홈페이지별 초기화면 수신 완료 시간을 나타낸다. 지연시간을 0.01초로 설정한 경우에는 A와 B 홈페이지의 수신 완료시간이 약 1초, C 홈페이지의 수신 완료시간은 약 4초였다. 만일 디코이 포트가 10개 생성된 클라이언트에 0.01초의 지연 시간을 적용할 경우 해커가 10개의 포트에 DOS 공격을 수행하면, 최악의 경우 초당 약 1MB를 전송하게 되어 시스템과 네트워크에 큰 부하를 줄 수 있다. 지연시간을 0.1초로 설정했을 경우, 홈페이지 초기 화면의 크기가 비교적 작은 A와 B 홈페이지는 브라우저가 수신 완료하는데 약 3초의 시간이 걸리며, 크기가 큰 C 홈페이지는 수신 완료하는데 20초가 걸렸다. 현재 인터넷 환경을 감안하여 웹페이지 수신 완료에 걸리는 시간이 20초나 걸릴 경우, 해커가 디코이 포트에 접근할 가능성은 낮

〔표 2〕 제안 시스템과 허니트랩의 시스템 자원 사용 비교

시스템 자원	제안 시스템	허니트랩
CPU	0.0 %	1%
메모리	0.1 %	10초마다 3.2%씩 증가
네트워크	11KiB/s	1.9MiB/s

아질 수 있다. 따라서 해커의 공격을 허용하기 위한 허니팟에 비교적 작은 웹페이지로 구성된 홈페이지를 구축하고 지연 시간을 0.1초로 설정하면, 초당 약 100KB만 전송 가능하기 때문에 클라이언트의 부하를 크게 줄일 수 있다.

[표 2]는 제안 시스템과 허니트랩이 파일을 포워딩할 때의 시스템 자원 사용량을 나타낸다. 시스템 자원 사용량 측정을 위해 700MB의 파일을 포워딩하였으며, 10초 마다 자원 사용 값을 측정하였다. 제안 시스템은 1024바이트의 데이터 전송 크기와 0.1초의 지연 시간에 따라 포워딩하도록 설정되었다. [표 2]에서와 같이 허니트랩은 패킷을 포워딩할 때 전송 속도를 제어하지 않기 때문에 클라이언트에게 커다란 부하를 준다. 메모리 사용량의 경우 10초마다 3.2%씩 증가하여 짧은 시간 내에 시스템이 매우 느려지는 현상이 일어났다. 그러나 제안 시스템은 허니트랩과 달리 지연 시간에 따른 전송 속도 제어를 통해 시스템의 부하를 최소화 한다.

[표 3]에서는 제안 시스템과 기존 시스템들의 특징을 비교하였다. 제안 시스템은 허니트랩과 달리 해커를 유인하기 위한 포트를 자동으로 생성하기 때문에 시스템 설정이 용이하며, 서버가 아닌 클라이언트에 설치되기 때문에 21, 23, 25, 80번 포트와 같이 서버 프로그램들에 사용되는 포트들을 해커 유인을 위한 목적으로 사용할 수 있는 가능성이 유인포트 시스템에 비해 높다. 그리고 실험을 통해 클라이언트를 보호할 수 있는 수준으로 패킷 포워딩을 수행하기 때문에 허니트랩과 비교하여 DOS 공격에 강인하다. 또한, 시스템 보호 기능이 전혀 없는 허니트랩과 네트워크 레벨에서 해커의 IP 주소를 기반으로 서버 접근을 제한하는 유인포트 시스템과는 달리 제안 시스템은 네트워크 레벨과 응용 레벨에서 침입을 탐지 및 차단하며, 설치 프로그램의 취약점으로 인해 발생할 수 있는 버퍼오버플로우나 포맷스트링과 같은 쉘 코드 공격을 차단할 수 있다. 마지막으로 제안 시스템은 기존 시스템과 달리 포워딩 되는 패킷 데이터의 변조 가능을 제공함으로서 설치 프로그램이나 허니팟의 존재를 숨길 수 있다.

Ⅵ. 결 론

대부분의 보안 시스템들은 시그너처를 기반으로 해커의 공격에 대응하고 있으며, 신속하고 정확한 시그너처 생성을 위해서 허니팟에 관한 연구가 활발히 진행중이다. 그러나 허니팟은 미리 예약된 IP 주소에 접근하는 행위를 공격으로 간주하기 때문에 공격 정보 수집에 제한이 따른다. 또한, 사용하지 않는 포트를 이용하여 해커들의 공격 정보를 수집하는 허니트랩과 유인포트 시스템은 실제 환경에 적용하기에 여러 문제점을 가지고 있다.

이에 본 논문에서는 기존 시스템들의 문제점을 보완하고 기능을 확장한 에이전트 기반의 공격 정보 수집 시스템을 제안하였다. 한정된 IP 주소를 기반으로 해커의 공격을 기다리는 허니팟과 달리 제안 시스템의 에이전트는 인터넷 상의 많은 클라이언트에 설치되어 해커를 유인하기 때문에 많은 공격 정보 수집이 가능하

〔표 3〕 기존 시스템과 제안 시스템 비교

	유인포트 시스템	허니트랩	제안 시스템
사용 포트 선정 방법	• 스캔을 통해 가용한 0-1023번 포트를 자동으로 선택 사용	• 사용자 수동 설정	• 스캔을 통해 가용한 0-49151번 포트를 자동으로 선택 사용
트래픽 제어	• 한계값 설정을 통한 밀리 초당 패킷량 제한	• 시간과 무관한 전체 전송량의 제한	• 클라이언트 자원을 보호할 수 있는 수준으로 패킷량 제한
시스템 보호	• IPTable을 이용한 해커의 접근 방어 • 트래픽 제어를 통한 DOS 방어	• 기능 없음	• IPTable과 NIDS 기능을 통한 시그너처 기반의 침입 탐지 및 차단 • 트래픽 제어를 통한 DOS 방어 • 커널 모듈을 통한 쉘 코드 공격 방어 • 패킷 데이터 변조 기능을 통한 시스템 노출 방어

다. 또한 사용하지 않는 포트를 기반으로 공격 정보를 수집하는 기존 시스템들과는 달리 에이전트의 존재 여부를 숨기기 위한 데이터 변조 기능을 제공하며, 해커가 목표를 공격하기에 만족스러우며 클라이언트의 자원을 보호할 수 있는 수준으로 동작한다. 실험에서 허니트랩은 대용량의 파일을 포워딩할 때 클라이언트에게 커다란 부하를 줬지만, 제안 시스템은 시스템 및 네트워크 과부하를 예방할 수 있었으며 자체 시스템에서 발생할 수 있는 취약점 공격을 차단할 수 있었다. 마지막으로, 제안 시스템은 매니저의 통합 관리 기능을 통해 에이전트의 시그너처 파일이 주기적으로 업데이트되기 때문에 새로운 공격에 민첩하게 대응하며, 클라이언트의 운영환경에 맞는 시스템 자동 설정을 통해 실제 환경에서 매우 유연하게 동작한다.

제안 시스템은 클라이언트 관점에서 자신의 컴퓨터를 보호하기 위한 보안 프로그램으로 동작하며, 보안 관련 연구기관에서는 해커를 유인하기 위한 덫으로 사용 가능하기 때문에 매우 유용하다 판단된다.

참고문헌

[1] Martin Roesch, "Snort-Lightweight Intrusion Detection for Networks," *Proceedings of the LISA*, 1999.

[2] Brian Laing, Jimmy Alderson, *How to Guide: Implementing a Network Based Inrusion Detection System*, Internet Security System, 2000.

[3] L. Spitzner, *Know Your Enemy: Sebek2 A Kernel Based Data Capture Tool*, http://www.honeynet.org, 2003.

[4] Xing-Yun He, Knok-Yan, Siu-Leung Chung, Chi-Hung Chi, Jia-Guang Sun, "Real-Time Emulation of Intrusion Victim in HoneyFarm," *Proceedings of the AWCC*, 3309, pp. 143-154, Nov 2004.

[5] Miyoung Kim, Misun Kim, Youngsong Mun, "Design and Implementation of the HoneyPot System with Focusing on the Session Redirection," *Proceedings of the ICCSA*, 3043, pp. 262-269, May 2004.

[6] John G. Levine, Julian B. Grizzard, Henry L. Owen, "Using Honeynets to Protect Large Enterprise Networks," *IEEE Security and Privacy*, 2, pp. 74-75, 2004.

[7] Tillmann Werner, *Honeytrap: Trap Attacks against TCP Services, http://honeytrap.sourceforge.net.*

[8] 김익수, 김명호, "사용되지 않는 포트를 이용하여 해커를 허니팟으로 리다이렉트하는 시스템 설계 및 구현," *한국정보보호학회논문지*, 16(5), pp. 15-24, October 2006.

[9] 이현우, 이상엽, 정현철, 정윤종, 임채호, "Analysis of Large Scale Network Vulnerability Scan Attacks and Implementation of The Scan-Detection Tool," 1999.

[10] *Know Your Enemy: Honeynets, http://www.honeynet.org*, 2005.

[11] L. Spitzer, *Honeypots: Tracking Hackers*, Addison-Wesley, 2002.

[12] *Know Your Enemy: Honeywall CDROM, http://www.honeynet.org*, 2004.

[13] *브라질 사이버테러 정보보호 현황 및 대응기구*, 국가사이버안전센터, Monthly 사이버 시큐리티 1월호.

[14] Cristine Hoepers, Klaus Steding-Jessen, Luiz E. R. Cordeiro and Marcelo H. P. C. Chaves, "A National Early Warning Capability Based on a Network of Distributed Honeypots," *17th Annual FIRST Conference on Computer Security Incident Handling*, 2005.

〈著者紹介〉

김 익 수 (Ik-Su Kim) 학생회원
2000년 2월: 숭실대학교 컴퓨터학부 졸업
2002년 2월: 숭실대학교 컴퓨터학과 석사
2002년 3월~현재: 숭실대학교 컴퓨터학과 박사과정
<관심분야> 정보보호, 시스템 보안, 인터넷 보안

김 명 호 (Myung-Ho Kim) 종신회원
1989년 2월: 숭실대학교 전자계산학과 졸업
1991년 2월: 포항공과대학교 전자계산학과 석사
1995년 2월: 포항공과대학교 전자계산학과 박사
1998년~1999년 University of Tennessee 전자계산학과 교환교수
1995년~현재 숭실대학교 컴퓨터학부 부교수
<관심분야> 병렬/분산처리, 컴퓨터 보안, BI, 클러스터링, 리눅스