액티브 네트워크 기반 침입자 역추적 메커니즘

The intruder traceback mechanism based on active networks

이 영 석*
Young-seok Lee

요 약

최근 인터넷을 통한 사이버 공격의 형태가 다양해지고 복잡해지면서 네트워크 침입자를 효과적으로 탐지하고 신속하게 대응하는 것이 어려워지고 있다. 특히, UDP 계열의 DoS, DDoS 공격에서 IP 패킷 내의 근원지 주소를 다른 호스트의 IP로 위조하여 공격하거나, TCP 기반의 우회 연결 공격 등의 경우에서 침입자의 실제 위치를 추적하고 고립화 시키는 것은 어려운 일이다. 본 논문에서는 TCP 연결 공격과 같은 사이버 공격에 신속히 대응하기 위해 액티브 네트워크에 기반한 보안 구조를 제안한다. 액티브 네트워크를 이용하여 보안 관리 프레임워크를 설계하고, TCP 기반 네트워크 침입자를 추적하고 고립화하기 위한 보안 메커니즘을 구현한다. 구현된 보안 메커니즘이 공격 시나리오에 따라 동작하는 과정을 실험 환경에서 시험하며, 실험 결과를 분석한다.

Abstract

Recently, the patterns of cyber attack through internet have been various and have become more complicated, and thus it is difficult to detect a network intruder effectively and to response the intrusion quickly. Therefore, it is almost not possible to chase the real location of a network intruder and to isolate the intruder from network in UDP based DoS or DDoS attacks spoofing source IP address and in TCP based detour connection attacks. In this paper, we propose active security architecture on active network to correspond to various cyber attacks promptly. Security management framework is designed using active technology, and security control mechanism to chase and isolate a network intruder is implemented. We also test the operation of the active security mechanism implemented on test_bed according to several attack scenarios and analyze the experiment results.

Keyword : DoS, DDoS, Active network, Traceback mechanism

1. 서 론

최근 인터넷 상에 발생하는 사이버 공격은 고도의 기술을 이용하여 점차 다양화되고 지능화되고 있으며, 이에 따라, 국가적으로 중요한 정보통신망에 대한 사이버 공격 위협이 증대하고 있다. 그 특징으로는 분산 환경에서 다수 공격 에이전트를 이용하여 특정 상용 서버의 서비스 제공을 마비시키는 분산 서비스 거부 공격의 출현과 해외 해커들의 국내 전산망을 우회 루트로 활용한 사례의 증가 등 사이버 공격 행위가 점차 범죄의 강력한 주요 수단으로 이용되는 추세에 있다. 이런 환경 변화에 따라 사이버 공격에 대해 기존 네트워크 보안에 비해 사용자 지향적이고, 능동적이며 좀 더 강력한 대응을 할 수 있는 네트워크 보안 기술의 개발 필요성이 대두되고 있다.

현재의 네트워크 보안 관리는 방화벽, 침입탐지시스템을 결합하여 자신의 도메인 상에서 어떻게 효과적으로 공격을 탐지하고, 그 공격 트래픽으로부터 해당 도메인을 보호할 것인가에 초점이 맞추어져 있다. 반면, 공격의 경우 서비스거부공격(DoS, Denial of Service)과 같이 연결 설정 없이 다량의 특정 서비스 요청 패킷을 대량 발송하는 UDP 계열의 공격은 요청 패킷 발송 시 소스 주소를 거짓 주소로 설정하게 된다. 또한, 시스템 상태 변경이나 정보를 획득하기 위해 특정 시스

* 정 회 원 : 군산대학교 전자정보공학부 교수
leeys@kunsan.ac.kr(제 1저자)
[2004/02/18 투고 - 2004/02/25 심사 - 2004/07/21 심사 완료]

템으로 세션이 연결되는 TCP 계열의 공격인 경우에도 공격 호스트로부터 목표 호스트로 직접 접속하여 공격하는 것이 아니라 여러 개의 경유 호스트를 거쳐서 목표 시스템으로 접속하게 된다.

따라서, 특정 공격이 탐지/차단되더라도 공격자는 네트워크에 대한 액세스를 계속적으로 유지할 수 있고, 이에 따라 다른 도메인으로의 또 다른 공격이나 동일 도메인으로도 다른 공격 기법을 적용하거나 경유 호스트를 달리하여 제 2, 제 3의 공격이 가능하다.

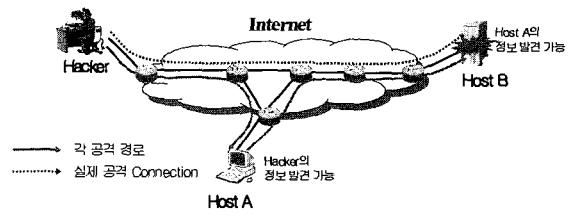
결국, 공격자의 네트워크에 대한 접속 자체를 차단하여 네트워크로부터 고립화시키는 것과 같은 능동적이고 강력한 대응을 위해서는 공격자의 실제 위치를 파악하는 것이 필요하다. 공격자 위치를 추적하기 위해 UDP 계열의 공격과 TCP 계열의 공격에 적용되는 기법이 달라질 수 있다. 일반적으로 네트워크 노드에서 라우팅 패킷에 대한 로그를 남길 경우 두 유형의 공격 모두에 대해서 추적이 가능하나, TCP 계열의 공격의 경우 호스트 사이에 TCP 세션 연결이 설정되고 유지되므로 이를 이용하는 것이 좀 더 효과적일 수 있다.

본 논문에서는 이러한 보안 환경 변화에 따른 요구사항을 반영하고 능동적인 대응을 수행할 수 있는 보안 구조로서 액티브 네트워크를 이용한 보안 관리 프레임워크를 제안하고자 한다. 제안된 보안 관리 프레임워크를 기반으로 TCP 연결 우회 공격과 같은 사이버 공격에 신속하고 능동적으로 대처하는 것이 가능할 뿐만 아니라, TCP 연결 우회 공격의 침입자를 추적하고 고립화하는 것도 가능하다.

본 논문의 구성은 다음과 같다. 2장에서는 TCP 연결 역추적을 위해 연구되고 있는 역추적 기술에 대해 소개한다. 3장에서는 침입자 추적 및 대응을 위해 설계된 보안 프레임워크가 기술되며, 4장에서는 보안 프레임워크 상에서 TCP 연결 역추적 및 대응 메커니즘이 기술되며, 5장에서는 시범 네트워크 상에서 제안된 메커니즘의 동작 과정에 대한 실험 과정을 기술하고 실험 결과를 분석한다.

2. 관련 연구

현재 사용되고 있는 대부분의 역추적 기법은 사람에 의해 진행되고 있는 실정이다. 이와 같이 사람에 의해 직접 해커의 흔적을 찾아내 역추적을 수행하는 경우, 해커의 실제 위치를 찾기 위해서는 공격을 당하고 있는 시스템(그림 1의 Host B)의 로그 기록뿐만 아니라, 공격을 수행하고 있는 시스템(그림 1의 Host A)의 로그 기록을 분석해야 한다[1]. 왜냐하면 Host B의 로그기록에서 얻을 수 있는 정보는 Host A에서 해킹이 시도되었다는 것뿐이기 때문이다. 그런데, 실제 해커의 위치는 Host A가 아니므로, 이를 파악하기 위해서는 Host A의 로그 기록을 확인해야 한다. 즉, 역추적 경로상에 존재하는 모든 중간 경유지에 대해 해커의 흔적을 조사하여야 한다. 그림 1에서는 경유 시스템이 1개뿐이지만 실제 상황에서는 다수의 경유 시스템이 존재할 수 있다. 따라서 다음과 같은 문제점들이 존재한다.



〈그림 1〉 경유 시스템을 이용한 공격 경로

Host A와 Host B의 거리가 지리적으로 먼 거리에 위치하여 직접 그 시스템을 확인 할 수 없는 경우라면, 실제 해커의 위치를 파악하는데 많은 시간과 노력이 필요하게 된다. 또한 수작업을 통한 역추적은 기술 인력의 부족으로 늘어나는 해킹 사고에 빠르게 대응하지 못하고 있다. 역추적을 수행하는 과정에서 중간 경유 시스템으로부터 공격 경로상의 이전 시스템에 대한 정보를 얻을 수 없는 경우, 역추적이 불가능하게 된다. 이는 역추적을 위하여 오직 로그 파일에만 의존하

기 때문이다. 이와 같은 문제점들 때문에, 신속한 역추적을 수행하기 위해 자동화된 시스템이 절실히 필요한 것이다[2].

역추적 기술은 일반적으로 크게 2가지 분야로 분류되며 해커가 우회공격을 시도하는 경우, 해커의 실제 위치를 추적하는 기술과 IP주소가 변경된 패킷의 실제 송신지를 추적하는 기술이다. 이때, 우회 공격을 시도하는 해커의 실제 위치를 추적하는 기술을 TCP 연결 역추적(TCP Connection Traceback) 혹은 연결 역추적(Connection Traceback)이라 하고, IP 주소가 변경된 패킷의 실제 송신지를 추적하는 기술을 IP 패킷 역추적(IP Packet Traceback) 혹은 패킷 역추적(Packet Traceback)이라 한다.

2.1 TCP 연결 역추적 기술

TCP 연결 역추적이란 TCP 연결을 기반으로 우회 공격을 시도하는 해커의 실제 위치를 실시간으로 추적하는 기법이다. 여기서 우회 공격이란, 그림 1에서 볼 수 있듯이, 해커(Hacker)가 최종 침입 대상(Host B)을 공격하는데 있어서, 직접 Host B로 침입을 시도하지 않고, 다수의 중간 시스템(Host A)을 경유하여 침입을 수행하는 공격을 의미한다. 이와 같이 해커가 우회 공격을 시도하게 되면, 직접 공격을 당하는 피해 시스템(Host B)에서는 해커의 실제 위치(Hacker)를 파악할 수 없게 된다. Host B에서 확인할 수 있는 것은 오직 현재 해당 시스템을 공격하고 있는 바로 이전의 시스템(Host A)에 대한 정보만을 얻을 수 있다.

이와 같은 TCP 연결 역추적 기술은 흔히 연결 체인(Connection Chain) 역추적 기술이라고 불리기도 한다[3].

TCP 연결 역추적 기술은 다시 2가지로 분류할 수 있다. 이는 호스트 기반 연결 역추적(Host-based connection traceback) 기술과 네트워크 기반 연결 역추적(Network-based connection traceback) 기술로 분류된다[4].

- 호스트 기반 연결 역추적(Host-based Connection Traceback) 기술
  호스트 기반 연결 역추적 기술은 역추적을 위한 모듈이 인터넷 상의 호스트들에 설치되는 역추적 기법으로 호스트에서 발생하는 로그 기록 등의 다양한 정보를 바탕으로 역추적을 진행하는 기술이다. 그러나 이러한 방법을 이용하여 역추적을 수행하기 위해서는 인터넷 상의 모든 호스트에 역추적 모듈이 설치되어야 하고, 역추적 경로 상의 단 1개의 시스템에서라도 어떤 문제에 의해서 역추적 정보를 얻을 수 없게 되는 경우가 발생하면 역추적이 불가능하게 되는 단점을 가지고 있다. 이와 같은 문제점들로 인해 현재의 인터넷 환경에 적용하는 것은 거의 불가능하다고 할 수 있다[5].

- 네트워크 기반 연결 역추적(Network-based Connection Traceback) 기술
  네트워크 기반 연결 역추적 기술은 네트워크 상에 송수신되는 패킷들로부터 역추적을 수행할 수 있는 정보를 추출하여 역추적을 수행하는 것으로 역추적 모듈이 네트워크 상에 송수신되는 패킷을 확인할 수 있는 위치에 설치된다. 현재 제안되고 있는 방법은 대부분 송수신 패킷을 확인할 수 있는 위치에서 공격 연결과 같은 연결 체인에 속하는 연결을 추출하여 역추적을 수행하는 방법을 취하고 있다.

그러나 아직까지 네트워크 기반 연결 역추적 기술을 현재의 인터넷에 적용하여 사용할 수 있는 전체 시스템은 제안되지 못했다. 다만 네트워크 상에서 얻을 수 있는 패킷으로부터 어떤 정보를 활용해야 공격 연결과 같은 연결에 속하는 가를 판단할 수 있을지에 대한 알고리즘만이 제기되고 있는 상황이다. 이는 네트워크 상의 패킷들로부터 얻게 되는 각종 연결 정보들을 네트워크 상에 존재하는 역추적 시스템들과 공유하는데 있어서 생성되는 정보의 순서관계 및 동기화가 매

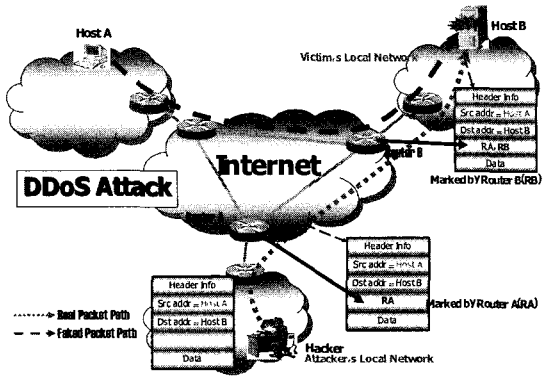
우 어렵고, 네트워크 상에서 발생하는 모든 연결에 대한 정보를 지속적으로 보유하고 있어야 하는 문제가 발생할 수 있기 때문이다.

또 다른 네트워크 기반 연결 역추적 기술로는 액티브 네트워크(Active Network) 상에서 동작하는 기술이 있으며[6], 본 연구에서는 액티브 네트워크를 기반으로 동작하는 역추적 기술을 설계하고 개발하였다. 그러나, 본 연구에서는 순수한 네트워크 기반 연결 역추적 방식이 아니라 호스트 기반 연결 역추적 방식을 액티브 네트워크에 적용한 새로운 역추적 모델을 제안하였다. 제안된 역추적 모델은 액티브 네트워크 상에서 동작하기 때문에, 이러한 역추적 방식은 현재의 인터넷 환경에 신속하게 적용하기 어려운 측면이 있다.

2.2 IP 패킷 역추적 기술

IP 패킷 역추적(IP Packet Traceback) 기술은 앞서 잠시 언급한 바와 같이 IP주소가 변경된 패킷의 실제 송신지를 추적하기 위한 기술을 말한다[7,8]. 일반적으로 IP 주소가 변경된 패킷은 악의적으로 사용되는 경우가 대부분이다.

특히 서비스 거부, 혹은 분산 서비스 거부(DDoS, Distributed Denial of Service) 공격에 주로 사용된다. IP 주소가 변경되는 경우에는 TCP 연결을 유지할 수 없기 때문에, 일방적인 패킷 송신으로 공격이 가능한 DoS 혹은 DDoS에서 주로 사용되는 것이다. 물론 과거 IP Spoofing이라 알려져 있는 해킹 기법을 이용하는 경우, IP 주소가 변경된 패킷을 이용하여 공격하고자 하는 대상 시스템에 백도어를 설치하도록 하는 기법이 사용되기도 하였으나, 이를 위해서는 TCP Sequence Number Guessing 과정이 필요하기 때문에 최근에는 거의 사용되지 않고 있다. 또한 IP 패킷 역추적은 현재 특정 시스템으로 IP주소가 변경된 패킷을 송신하는 시스템을 찾는 기술로서, 여러 중간 경유지를 추적하여 실제 해커의 위치를 찾는 TCP 연결 역추적 기술과는 해결하고자



〈그림 2〉 IP 패킷 역추적

하는 문제의 대상에 약간의 차이가 있다.

IP 패킷 역추적 기법으로는, 그림 2에서 볼 수 있듯이 공격자(Hacker)가 전송하는 패킷에 해당 패킷을 전달한 라우터를 표시함으로써 추적할 수 있게 하는 패킷 표시 기법[9]을 이용한 연구와 다른 여러 기법을 통한 IP 패킷 역추적을 위한 연구가 진행중이다[10,11].

3. 침입자 역추적을 위한 보안 관리 프레임워크

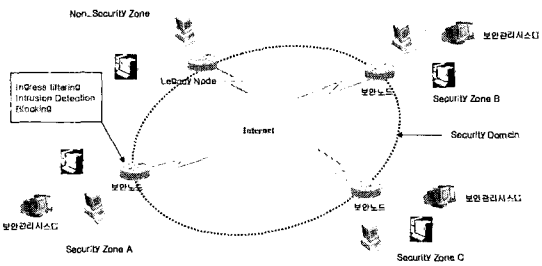
본 논문에서 설계된 보안 관리 프레임워크는 액티브 네트워크 기반으로 구성되며, 2장에서 설명된 호스트 기반 역추적 방식에 따라 로그 분석을 통하여 TCP 연결 공격에 능동적으로 대응하게 된다. 호스트 기반 역추적 방식을 액티브 네트워크에 적용할 경우 네트워크 보안이 능동적으로 발전할 수 있는 이유는 크게 다음과 같다[6].

- 네트워크 상의 라우터나 스위치들은 자신에게 전송된 보안 대응 프로그램을 특별한 프로토콜이 없이도 네트워크 처리 단계에서 인식하고 수행할 수 있다.
- 보안 관리자는 각각의 노드에서 필요한 보안 대응 수단을 프로그래밍에 의해 제어 할 수 있

다. 즉 네트워크 상에서의 보안 메커니즘을 보다 유동적이고 유연하게 배치, 삭제할 수 있으며, 다수의 네트워크 장비 및 보안장비를 분산 관리할 수 있다.

액티브 네트워크 기반의 보안 기법은 제품 개발시 탑재된 보안 기능을 정해진 프로토콜에 의해 상호 연동시키며, 제한된 범위에서의 보안 기능만을 제공할 수 있었던 기존의 정적인 보안 메커니즘의 문제들을 해결할 수 있는 새로운 개념의 보안 메커니즘이다.

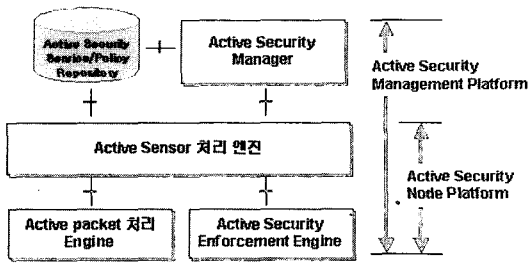
액티브 네트워크를 이용한 보안 프레임워크에서는 프로토콜 대신 액티브 패킷을 이용하며, 보안 기능 자체를 액티브 패킷 내의 보안 대응 프로그램으로 전송, 실행함으로써 보안 구조를 단순화시키고 네트워크 자원을 절약한다.



〈그림 3〉 액티브 네트워크 기반 보안 관리 프레임워크

본 논문에서 제안한 액티브 네트워크 기반의 보안관리 프레임워크는 그림 3에 도시한 바와 같이 가입자 네트워크 상에 액티브 네트워킹 기능을 제공하는 보안노드와 이들을 제어하는 보안관리시스템으로 구성되며, 논리적인 하나의 보안 관리 도메인을 형성한다. 또한 각 보안 관리 도메인은 전체 네트워크 상에 분산적으로 배치되며 상호간의 연동 및 협업을 위한 별도의 관리 계층은 갖지 않는다. 즉, 모든 보안 제어는 액티브 패킷을 통해 이루어지며 보안 도메인 간의 상호 연동과 협업 역시 액티브 패킷에 의해 수행된다.

각 보안 관리 영역은 전체 네트워크 상에 분산



〈그림 4〉 보안 프레임워크 시스템 플랫폼

적으로 배치되어 상호간의 연동 및 협업을 수행하며 이를 위한 별도의 관리 계층은 갖지 않는다.

보안노드에는 그림 4와 같이 보안 센서를 수신하고 실행시킬 수 있는 센서 처리 엔진이 탑재되며, 보안관리시스템에는 보안 관리를 위한 어플리케이션이 추가적으로 탑재된다. 보안 센서는 액티브 패킷 내에 포함된 TCP 연결 역추적을 위한 실행 프로그램으로서, 보안노드와 보안관리시스템 상에서 수행된다. 액티브 패킷 내에 포함되어 전달되는 보안 센서에 TCP 연결 역추적을 위한 전체 실행 프로그램이 삽입된다면 액티브 패킷의 전체 크기가 커져 전송 시 문제점이 발생한다. 따라서, 보안 센서의 실제 실행 프로그램은 별도의 저장소를 통해 저장하고 보안 센서에는 실행 프로그램의 식별 정보만을 저장한다.

액티브 네트워크에서 보안 센서를 이용하여 네트워크의 보안 상태를 관리하기 위해서는 보안관리 도메인 내의 각 보안 시스템에 보안 센서를 인지하고 실행시킬 수 있는 기능이 탑재되어 있어야 한다. 보안노드 및 보안관리시스템에는 보안 센서를 수신하고 실행시킬 수 있는 보안 센서 처리 블록이 공통적으로 탑재된다. 또한, 보안관리시스템에는 보안 관리를 위한 기능 블록과 보안 센서 및 정책을 관리하기 위한 저장소가 추가적으로 탑재되며, 보안노드에는 보안 센서에 의해 네트워크 차원의 실시간 공격 대응 기능을 제공하는 대응 블록이 추가된다.

보안 관리 서비스에 따라 관리자에 의해 생성

되는 보안 센서를 저장하고 관리하기 위한 데이터베이스관리시스템으로 디렉토리 서버를 이용한다.

보안 센서는 액티브 네트워크 상에서 보안 기능을 수행하는 일종의 액티브 패킷이다. 이러한 센서는 네트워크 침입에 능동적으로 대응하기 위한 소프트웨어 모듈로써, 액티브 패킷 내에서 실행 가능한 코드 형식으로 전달된다. 보안 센서는 보안노드와 보안관리시스템에서 수행되며 센서의 데이터를 변경할 수 있고 다른 보안노드나 보안관리시스템으로의 이동성을 갖는다.

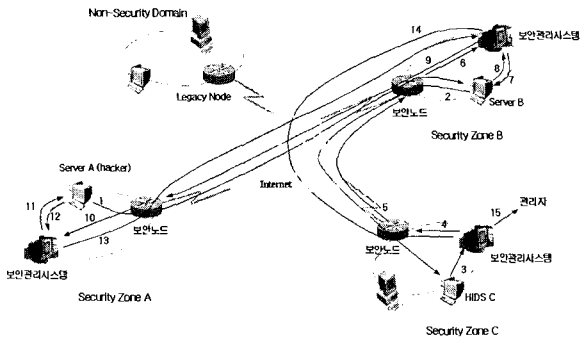
센서는 또한 보안 관리를 위해 TCP 세션 추적 기능 이외에 위조 IP 추적 기능, DDoS 추적 기능 등을 수행하기 위한 데이터도 제공가능하다. TCP 세션 역추적을 위한 보안 센서의 종류 및 기능은 다음과 같다.

- TCP Session Tracing 센서 : 침입자의 TCP 세션 연결을 역추적하여 침입자의 위치 를 파악하고 해당 TCP 세션을 제거하며 침입자 호스트를 네트워크로부터 고립시키는 기능을 수행
- TCP Tracing Complete 센서 : 역추적 결과를 보안관리시스템에게 전달하는 기능을 수행

4. TCP 세션 역추적을 위한 보안 메커니즘

TCP 연결 공격은 telnet, rlogin, FTP와 같은 TCP 계열의 리모트 접속 서비스를 이용하여 여러 호스트들을 경유한 침투를 시도함으로써 침입자의 위치를 위장한 후 공격을 시도하는 우회 공격이다.

본 연구에서 제안된 액티브 네트워크 기반의 TCP 연결 공격 역추적 방식에서는 효율적인 역추적을 위해 경유지 보안노드와 경유지 도메인 내의 보안관리시스템과의 연계가 필수적이다. 따라서, 본 연구에서는 호스트에서 발생하는 TCP 연결 세션을 감시하고 세션에 대한 기록을 남기는 TCP 세션 로그 에이전트를 각 호스트에 상주시키고, 보안관리시스템에는 세션 로그 매니저를 탑재하여 특정 세션에 대한 과거 접속 정보에 대한 추적이 가능하도록 설계하였다.



〈그림 5〉 TCP 연결 역추적 및 대응 과정

그림 5는 본 연구에서 설계된 TCP 연결 공격에 대한 역추적 과정과 대응, 그리고 역추적의 결과를 통보하는 메커니즘을 단계별로 나타낸 것이다.

(단계 1) 보안 영역 A 내의 서버 A를 사용하는 해커는 보안 영역 B 내의 서버 B에 접속하여 보안 영역 C 내의 HIDS_C로 침입하기 위한 경유지 호스트로 사용한다. 서버 B의 로그 매니저는 모든 접속 기록을 기록을 저장한다.

(단계 2) 서버 B에 접속한 침입자는 HIDS_C로 TCP 연결 공격을 시도한다.

(단계 3) HIDS_C는 서버 B로부터의 TCP 연결 공격을 탐지하고, 침입 사실을 보안관리시스템 C로 통보한다.

(단계 4) 보안관리시스템 C는 TCP Session Tracing 센서를 생성하여 역추적을 수행한다. TCP Session Tracing 센서는 액티브 패킷 내에 포함되어 HIDS_C를 공격한 서버 B를 목적지로 하여 전송된다. 보안노드 C는 먼저 액티브 패킷을 수신하고 패킷 내에 포함된 TCP Session Tracing 센서를 수행한다. 일단 TCP Session Tracing 센서의 실행 코드가 코드 서버로부터 보안노드 C의 메모리에 적재되며 TCP Session Tracing 센서 실행 코드에 의해 보안노드 C는 서버

B(source IP)로부터 HIDS_C(destination IP)로 입력되는 패킷들을 차단한다.

(단계 5) 그런 다음 보안노드 C는 수신된 TCP Session Tracing 센서를 재전송한다. 보안노드 C에 의해 재전송된 액티브 패킷은 목적지 주소(서버 B)에 따라 라우팅되어 보안노드 B가 수신한다.

(단계 6) 보안노드 B는 TCP Session Tracing 센서를 수신한 후, TCP Session Tracing 센서 실행 코드를 코드 서버로부터 적재한다. 보안노드 B는 TCP Session Tracing 센서 실행 코드에 따라, 서버 B(source IP)으로부터 HIDS_C(destination IP)로 나가는 패킷들을 차단한다. 패킷 차단 이후에 TCP Session Tracing 센서를 포함한 액티브 패킷의 목적지 주소를 서버 B에서 보안관리시스템 B로 변경한 후, 전송한다.

(단계 7) 보안관리시스템 B는 TCP Session Tracing 센서를 수신하고 센서 내의 정보를 분석한다. 분석 결과에 따라, 보안관리시스템 B는 서버 B에서 수행되는 로그 매니저에게 TCP 연결 분석을 요청한다.

(단계 8) 분석 요청에 의해 서버 B는 HIDS_C로 TCP 연결이 되어 있는 세션이 어디로부터 접속되었는 지를 파악한다. 세션 분석 요청을 받은 서버 B 내의 로그 매니저는 해당 TCP 세션을 분석하여 서버 A으로부터 TCP 연결이 되었다고 판단한다. 서버 B의 로그 매니저는 TCP 세션 분석을 요청한 보안관리시스템 B로 분석 결과를 반환한다.

(단계 9) 보안관리시스템 B는 로그 매니저 분석 결과를 기반으로 TCP Session Tracing 센서의 송신자가 자신이라는 정보를 포함하여 새로운 TCP Session Tracing 센서를 생성하고, 서버 A를 목적지로 TCP Session Tracing 센서를 전송한다. 이 액티브 패킷은 다시 보안노드 B에 의해 수신되며, 보안노드 B는 TCP Session Tracing 센서를 수신한 후, 서버 A(source IP)으로부터 서버 B(destination IP)으로 입력되는 패킷을 차단한다. 패킷 차단 이후, 보안노드 B는 수신된 TCP Session Tracing 센서를 보안노드 A로 재전송한다.

(단계 10) 보안노드 A는 수신된 액티브 패킷 내의 TCP Session Tracing 센서를 실행하기 위해 실행 코드를 코드 서버로부터 적재한다. 보안노드 A는 TCP Session Tracing 센서 실행 코드에 의해 서버 A(source IP)에서 서버 B(destination IP)으로 나가는 패킷을 차단한다. 그런 다음 보안노드 A는 수신된 액티브 패킷의 목적지 주소를 서버 A에서 보안관리시스템 A로 변경하고 TCP Session Tracing 센서가 포함된 액티브 패킷을 보안관리시스템 A로 전송한다.

(단계 11) 보안관리시스템 A는 TCP Session Tracing 센서를 수신하고, 서버 A에서 수행되는 로그 매니저에게 서버 B로 접속한 TCP 연결 세션 분석을 요청한다.

(단계 12) 보안관리시스템 A는 서버 A의 로그 매니저로부터 서버 A에서 최초 연결된 TCP 세션이라는 분석 결과를 수신한다.

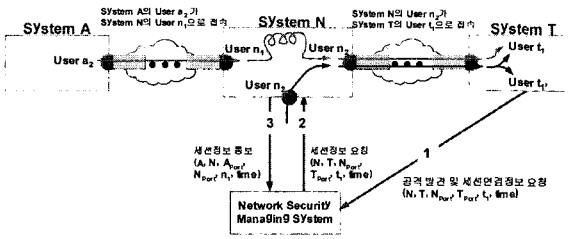
(단계 13) 보안관리시스템 A는 이러한 결과를 TCP Tracing Complete 센서에 저장하고, 수신된 TCP Session Tracing 센서 내의 정보에 따라 보안관리시스템 B를 목적지로 하여 TCP Tracing Complete 센서를 전송한다. 전송 경로에 의해, 보안 노드 A가 TCP Tracing Complete 센서를 포함한 액티브 패킷을 수신한다. 보안노드 A는 TCP

Tracing Complete 센서에 포함된 정보와 실행 코드에 따라, 보안노드 A는 서버 A(source IP)으로부터 외부망(destination IP)으로 나가는 모든 패킷을 차단한다. 그런 다음, TCP Tracing Complete 센서를 전송한다. TCP Tracing Complete 센서를 포함한 액티브 패킷은 전송 경로에 따라 보안노드 B에 의해 수신된다. 보안노드 B는 코드 서버로부터 TCP Tracing Complete 센서의 실행 코드를 적재한다. 해커가 경유한 경유지 도메인에 속한 보안노드 (예, 보안노드 B)에서는 TCP Tracing Complete 센서를 수행하지 않고 전송하며, 보안관리시스템 B가 이 센서를 최종 수신한다.

(단계 14) 보안관리시스템 B는 TCP Tracing Complete 센서 내의 분석 결과를 확인하고, 맨 처음 TCP Session Tracing 센서를 전송한 보안관리시스템 C로 TCP Tracing Complete 센서를 전송한다. 보안노드 B는 수신한 TCP Tracing Complete 센서를 수행하지 않고 전송한다. 보안노드 B가 전송한 TCP Tracing Complete 센서의 전달 경로에 따라, 보안노드 C가 TCP Tracing Complete 센서를 수신한다. 보안노드 C는 코드 서버로부터 TCP Tracing Complete 센서의 실행 코드를 적재한다. 피해 도메인에 속한 보안노드(예, 보안노드 C)에서는 TCP Tracing Complete 센서를 수행하지 않고 전송한다. 최종적으로, TCP Tracing Complete 센서는 TCP Session Tracing 센서를 전송한 보안관리시스템 C로 전달된다.

(단계 15) 보안관리시스템 C는 TCP Session Tracing 센서를 수신한 후, 센서 내에 포함된 정보를 분석하여 관리자에게 전달한다. TCP Session Tracing 센서나 TCP Tracing Complete 센서에 의해 보안노드에서 패킷을 차단하는 것은 관리자의 정책에 따라 해제될 수 있다. 차단 해제는 관리자의 영역으로 남는다.

본 연구에서는 TCP 연결 공격을 시도하는 공격자가 경유지로 사용한 호스트에는 로그 분석 매니저가 설치되어 수행하는 것을 가정한다. 제안된 TCP 연결 역추적 모델이 실제 인터넷 환경에서 적용된다면 TCP 연결 공격의 경유지로 사용될 가능성이 있는 모든 호스트에 로그 매니저를 설치해야 하므로 이러한 가정은 본 연구의 제약점으로 남는다. 그림 6은 경유지 서버(B)에 설치되어 있는 로그 매니저에서 공격자 세션을 추적하기 위한 로그 분석 메커니즘의 개념도를 보여준다.



〈그림 6〉 세션 역추적을 위한 로그 분석 메커니즘

그림 6에서 보듯이, 특정 호스트로(System T)부터 역추적을 위한 TCP 연결 정보를 요청받게 되면 보안관리시스템(Network Security Managing System)에서는 해커의 경유지 호스트(System N)로 해커의 TCP 연결 정보를 요청하고 해당 호스트(System N)는 보안관리시스템으로 요청받은 TCP 연결 정보를 로그에서 검색하여 전달한다.

TCP 연결 추적을 위해 각 호스트에서는 해당 호스트에 로그인한 TCP 연결과 타 호스트로 접속해 나간 TCP 연결 간의 매핑 정보를 관리하고

저장하는 기능을 수행한다. 이를 위해 송수신 패킷으로부터 추출된 정보와 해당 호스트에서의 사용자 작업 내역에 대한 로그를 남기고 이 두 데이터를 결합하여 해당 호스트에서의 입출력 TCP 연결에 대한 로그를 남긴다. 또한 외부 시스템과의 인터페이스를 위해 암호화된 통신 기능을 구현한다.

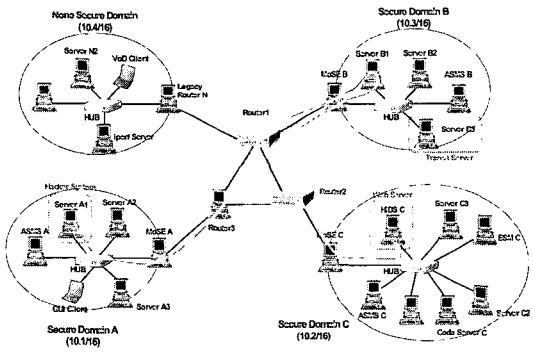
이 때, 공격이 탐지되면 보안관리시스템은 공격 TCP 연결을 구성하는 이전 호스트에 해당 TCP 연결 정보를 요청하게 되고 그 응답으로 이전 호스트 정보를 받게 된다. 이런 절차를 공격자 위치를 확인할 때까지 공격 TCP 연결을 구성하는 각 호스트에 대해 반복적으로 수행하게 된다.

5. 구현 및 실험결과

본 연구에서는 보안노드와 보안관리시스템을 Linux kernel version 2.4 상에서 구현하였고, TCP Session Tracing 센서와 TCP Tracing Complete 센서는 자바 언어를 이용하여 실행 프로그램을 구현하였으며, 센서를 실행하기 위한 실행 환경으로서 SUN Java Virtual Machine version 1.4.2를 이용하였다. 보안관리시스템에서 센서 수행 정책을 관리하기 위한 데이터베이스로서 PostgresSQL JDBC를 사용하였고, 센서 실행 프로그램을 저장한 저장소는 Netscape 사의 iPlanet Directory Server version 5.1을 사용하였다. 보안노드 상에서 유해 패킷의 필터링을 위해 libpcap 라이브러리(iptables 명령)를 사용하였다.

TCP 연결 공격이 탐지되면, 보안관리시스템에서 TCP Session Tracing 센서를 액티브 패킷 내에 저장하여 보안노드로 전달한다. 이 경우 액티브 패킷 내에 실제 센서 실행 프로그램을 모두 포함하게 되면 액티브 패킷의 크기가 너무 커지기 때문에, 네트워크 대역폭에 따라 송신측에서의 패킷 분할(fragmentation)과 수신측에서의 패킷 결합(assembly)이 수행되며 이에 따른 성능 저하가 발생된다. 따라서, 본 연구에서는 액티브 패킷 내에 포함된 보안 센서에는 센서의 식별 정보만을 포함하고 실제 센서를 실행하는 경우에, 보안노드와 보안관리시스템은 센서 저장소인 디렉토리 서버로부터 해당 센서 실행 프로그램을 다운로드 받는다. 보안노드 혹은 보안관리시스템으로 다운로드된 보안 센서는 자바 가상머신 상에서 해당 프로그램 실행 절차에 따라 수행된다.

본 연구에서 개발한 보안 메커니즘은 그 적용 범위가 광역 인터넷이라는 점에서 개발된 기능을 네트워크에 적용하고 검증하기가 매우 어렵다. 특히, 단일 로컬 네트워크 상에서는 개발된 기능을 검증할 방법이 없으므로, 제안된 보안 관리 프레임워크의 적용 가능성 및 보안 기능 검증을 위한 테스트베드를 구축하여 개발된 기능을 검증하였다.



〈그림 7〉 테스트 베드 구성 및 TCP 연결 공격 경로

그림 7은 4장에서 기술된 보안 메커니즘을 시험하기 위해 구성된 테스트베드를 보여준다. 중앙에 위치한 1개의 네트워크 도메인은 공중망의 역할을 하는 가상 ISP(Internet Service Provider) 도메인이고, 나머지 4개의 도메인은 ISP에 연결되어 있는 로컬 네트워크 도메인으로 공격자가 존재하는 네트워크 및 우회 공격 서버 또는 직접적인 피해를 입는 서버가 존재하는 피해자 네트워크로 이용된다.

테스트베드 시스템을 구성하는 각 주요 장비는 다음과 같다. ISP 경계(Edge) 라우터는 보안 관리 영역을 상호 연결하고, 가상 인터넷 환경을 구축

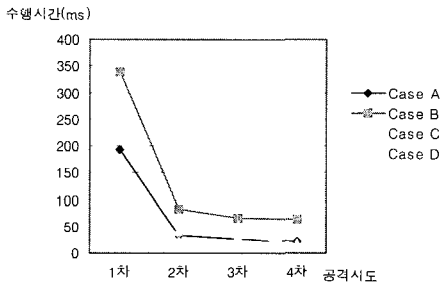
하기 위한 Core 네트워크를 구성하기 위해 사용되는 라우터 시스템이다. 보안관리시스템(ASMS, Active Security Management System)은 각 보안 관리 영역에 구축된다. 액티브 패킷을 처리할 수 있도록 확장된 리눅스 기반 커널 상에서 동작하는 시스템이며, 보안 센서 정보를 관리하기 위한 데이터베이스(PostgreSQL)가 탑재된다. 보안노드(MoSE, Mobile Sensor Engine)는 각 보안 관리 영역의 접속점에 구축된다. 액티브 패킷을 처리할 수 있는 확장된 리눅스 기반 커널로 동작하는 라우터이며, 패킷 차단, 프레임 모니터링(MAC, ARP 테이블 관리) 등 보안 대응 기능이 탑재된다. 가상 공격자 및 피해 시스템은 가상 해커를 가정하는 공격용 시스템과 공격 목표가 되는 시스템으로써 리눅스, SUN, Windows 등 다양한 플랫폼으로 구성된다.

TCP 연결 공격은 일반적으로 5~6개의 경유 호스트를 거쳐 공격 대상 서버에 침입하는 것으로 알려져 있다. 경유 호스트에서 공격자의 세션을 역추적하기 위한 기법은 경유 호스트의 수와는 무관하고, 테스트베드의 구성 상 1개의 경유 호스트를 거치는 것으로 하였다. 공격 대상 호스트는 호스트용 침입탐지시스템(HIDS)를 설치한 일반적인 웹 서버를 구축하여 홈 페이지를 통한 일반 웹 서비스를 제공한다.

TCP 확장 연결 공격 및 대응 절차는 다음과 같다.

- 서버 A1에서 우회 서버 B1을 통해 HIDS가 설치된 목표 시스템 HIDS_C로 로그인
- 목표 시스템에 설치되어 운용 중인 웹서버의 특정 페이지의 정보 변경
- HIDS_C가 해커에 의한 침입 발생 확인
- ASMS_C로 해커의 침입 통지
- 각 보안노드 및 보안관리시스템의 보안 메커니즘 구동 및 해커 침입 차단
- 보안 관리자에 의한 침입 차단 확인

그림 8은 TCP 확장 연결 공격 시에, 설계된 보안 프레임워크에서 보안 센서를 이용한 대응 메커니즘의 보안센서 수행 시간을 보여준다. 보안센서 수행시간은 공격 탐지 이후에, 보안노드 혹은 보안관리시스템에서 액티브 패킷 내에 포함된 보안센서를 수신한 시간을 기준으로 센서 저장소로부터 해당 센서 프로그램을 다운로드 받는 시간과 센서 실행 시간, 그리고 수신된 보안센서를 다시 액티브 패킷 내에 포함하여 재전송하는 시간의 합계로 측정된다. 한번 보안센서가 다운로드 되면 그 이후에는 다시 다운로드 되지 않는다. 따라서, 첫 번째 공격 이후에는 센서 프로그램 다운로드 시간이 포함되지 않기 때문에 공격 시도에 따른 센서 수행시간이 적다. Case A는 그림 9에서 피해 시스템이 속한 도메인 C의 경계에 위치한 보안노드(C)에서 수행된 TCP Session Tracing 센서의 수행 시간을 의미하며, Case B는 해커가 속한 도메인 A의 경계에 위치한 보안노드(A)에서 수행된 TCP Session Tracing 센서의 수행 시간을 의미한다. Case C와 Case D는 보안노드(C)와 보안노드(A)에서 수행된 TCP Session Tracing 센서의 수행시간을 의미한다.

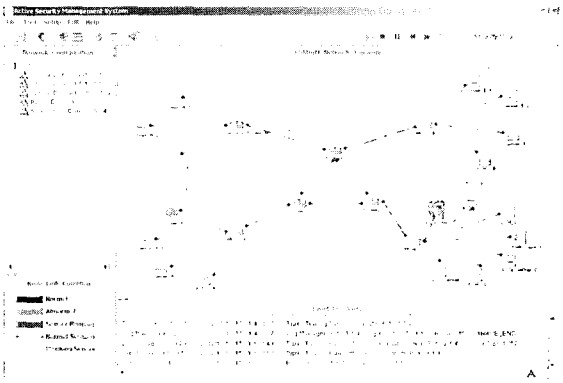


〈그림 8〉 TCP Session Tracing 센서 및 TCP Tracing Complete 센서 수행 시간

그림 8에서 보듯이, 보안노드(C,A)의 TCP Session Tracing 센서 수행 시간인 Case A와 Case B의 경우에서 해커의 1차 공격 시도인 경우는 1차 이후의 공격에서보다 수행시간이 길다. 이것은

보안 센서의 실행 프로그램이 저장된 센서 저장소(LDAP 서버) 즉, 프로그램 서버와 연결을 설정하고 실행 프로그램을 다운로드 하는 시간을 포함하기 때문이다. 특히, 해커가 이용 중인 서버 A1의 주소가 위조되었는지를 확인하기 위해 Case B의 경우는 보안노드(A)에서 TCP Session Tracing 센서를 수신한 후, ARP를 이용하여 실제 MAC 주소를 검증하는 오버헤드와 서버 A1으로부터 나가는 패킷들을 블록킹하는 기능이 포함되기 때문에 수행 시간이 가장 길다. 반면에, Case C와 Case D의 경우, 보안노드(C,A)에서는 TCP Tracing Complete 센서를 수신하지만 이미 연결되어 있는 코드 서버로부터 TCP Tracing Complete 센서를 다운로드 하며, 별도의 실행 없이 TCP Tracing Complete 센서를 재전송한다. 따라서, 이 경우에서의 보안 센서 수행시간은 아주 작다.

보안노드(B)에서의 센서 수행시간은 보안노드(C)에서의 센서 수행시간과 유사하므로 그림 9의 Case A와 C로 대체 가능하다.



〈그림 9〉 TCP 연결 공격에 대한 대응 결과

그림 9는 시험 절차 완료 후 시험 진행 상태를 보여 주기 위해 구성된 관리자 GUI 화면을 통해 공격자의 추적 및 차단이 완료된 상태임을 보여주고 있다. 그림에서 x 표시된 선은 공격자를 고립화시키기 위해 기존의 연결된 TCP 세션의 단절을 의미한다.

6. 결 론

본 논문에서는 네트워크 보안 환경 변화에 따르는 요구사항을 반영할 수 있는 확장된 보안 구조로서 액티브 네트워크를 이용한 보안 관리 프레임워크를 설계하였고, TCP 확장 연결 공격에 대응하기 위한 메커니즘을 제안하였다. 제안된 보안 관리 프레임워크는 TCP 세션 역추적 뿐만 아니라 위조 IP 추적 기능이나 DDoS 추적 기능도 제공할 수 있다. 또한, 현재의 네트워크 보안 메커니즘에 비해 전체 네트워크 차원에서 공격자에 대해 강력한 대응을 수행할 수 있고, 새로운 공격 기술, 방어 기술의 등장이나 보안 환경 변화에 유연하게 적응할 수 있다.

본 논문에서 제안된 보안 관리 프레임워크는 보안노드와 보안관리시스템으로 구성되며, 전체 네트워크 수준에서 수행할 보안 기능을 보안 센서 형태로 수행하도록 설계되었다. 설계된 보안 관리 프레임워크의 적용 가능성을 증명하기 위해 테스트베드를 구축하고, 해당 테스트베드 상에서 실제 제공되는 서비스와 대표적인 공격 기법을 적용한 상태에서 시스템의 보안 기능을 시험하였다. 시험 결과 제안된 보안 관리 프레임워크는 기존의 수동적인 침입 차단 및 침입탐지 시스템의 문제점을 해결하고 보다 능동적인 보안 서비스를 제공함을 보였고, 실제 필드에 적용할 수 있음을 확인하였다.

본 논문에서 제안된 보안 기술이 실제 네트워크에 적용된다면 현재의 네트워크 보안 기술이 제공하는 보안 수준 보다 한층 더 강력하고 광범위한 대응이 가능한 보안 수준을 확보할 수 있을 것이다.

참 고 문 헌

[1] 정현철, "Unix 로그 분석을 통한 침입자 추적 및 로그 관리 Part II", Korea Computer Emergency Response Team Coordination

Center 기술문서, 7. 2001.

[2] Chaeho Lim, "Semi-Auto Intruder Retracing Using Autonomous Intrusion Analysis Agent", FIRST Conference on Computer Security Incident Handling & Response 1999.

[3] K. Yoda and H. Etoh, "Finding a Connection Chain for Tracing Intruders", In F. Guppens, Y. Deswarte, D. Gollamann, and M. Waidner, editors, 6th European Symposisum on Research in Computer Security - ESORICS 2000 LNCS -1985, Toulouse, France, Oct 2000.

[4] 최양서, 서동일, 손승원, "역추적 기술 동향 : TCP Connection Traceback", 주간기술동향, 제1079호, 2003. 1.

[5] Y. Zhang and V. Paxson, "Detecting Stepping Stones", Proceedings of 9th USENIX Security Symposium, August 2000.

[6] 이수형, 나중찬, 손승원, "액티브 네트워크 기반 보안기술 동향", 주간기술동향, 제1076호, 2002. 12.

[7] Dawn X. Song and Adrian Perrig, "Advanced and Authenticated Marking Schemes for IP Traceback", Proceedings of InfoCom 2001.

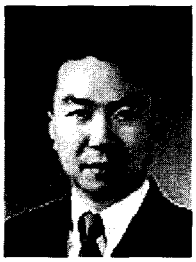
[8] Stefan Savage, David Wetherall, Anna Karlin "Practical Network Support for IP Traceback", Proceedings of the 2000 ACM SIGCOMM Conference, pp295-306, Stockholm, Sweden, Aug. 2000.

[9] Buchholz, Thomas E. Daniels, Benjamin Kuperman, Clay Shields, "Packet Tracker Final Report", CERIAS Technical Report 2000-23, Purdue University, 2000.

[10] D. Schnackenberg, K. Djahandari, and D. Sterene, "Infrastructure for Intrusion Detection and Response", Proceedings of DISCEX, January 2000.

[11] D. Schnackenberg, K. Djahandary, and D. Strene, "Cooperative Intrusion Traceback and Response Architecture(CITRA)", Proceedings of the 2nd DARPA Information Survivability Conference and Exposition(DISCEXII), June 2001.

◐ 저 자 소 개 ◑



이 영 석(Lee Young-seok)

1992년 충남대학교 컴퓨터공학과 졸업(학사)
1994년 충남대학교 대학원 컴퓨터공학과 졸업(석사)
2002년 충남대학교 대학원 컴퓨터공학과 졸업(박사)
1994년 ~ 1997년 LG전자정보통신연구소 연구원
2002년 ~ 2004년 한국전자통신연구원 선임연구원
2004년 ~ 현재 군산대학교 전자정보공학부 교수
관심분야 : 시스템/네트워크보안, 이동컴퓨팅, 컴퓨터네트워크 etc.
E-mail : leeys@kunsan.ac.kr