네트워크 시뮬레이터 도구를 이용한 침입자 역추적 알고리즘 검증

[A Verification of Intruder Trace-back Algorithm using Network Simulator (NS-2)]

서 동 일 † 김 환 국 †† 이 상 호 †††
(Dong-il Seo) (Hwan-kuk Kim) (Sang-ho Lee)

요 약 인터넷은 이미 일상생활에 깊게 자리 잡았다. 이러한 인터넷의 적용으로 실생활에서 수행하여야만 했던 많은 일들을 인터넷을 통해 수행할 수 있게 되었고, 인터넷의 편리함 때문에 인터넷 사용자 역시 크게 증가하였다. 그러나 인터넷 사용자의 증가와 더불어 인터넷을 통한 각종 침해사고 역시 크게 증가하였다. 침입자 역추적 기술은 여러 경유 시스템을 통해 우회 공격을 시도하는 해커의 실제 위치를 실시간으로 추적하는 기술이다. 따라서, 본 논문에서는 현재 인터넷 환경에 적용 가능한 워터마크 기법을 응용한 TCP 연결 역추적 시스템인 RTS 알고리즘을 제안한다. 그리고 제안된 역추적 알고리즘의 특징을 분석하여 역추적 구성 요소를 모델링하고, 네트워크 시뮬레이터 도구인 ns-2를 이용하여 모델링 된 역추적 구성 요소를 포함한 가상 네트워크 토폴로지 환경에서 모의 실험한 내용과 그 결과를 분석하였다.

키워드 : 네트워크 보안, 역추적, 해킹, 시뮬레이션, 정보 보호

Abstract *Internet has become an essential part of our daily lives. Many of the day to day activities can already be carried out over Internet, and its convenience has greatly increased the number of Internet users. But as Internet gains its popularity, the illicit incidents over Internet has also proliferated. The intruder trace-back technology is the one that enables real time tracking the position of the hacker who attempts to invade the system through the various bypass routes. In this paper, the RTS algorithm which is the TCP connection trace-back system using the watermarking technology on Internet is proposed. Furthermore, the trace-back elements are modeled by analyzing the proposed trace-back algorithm, and the results of the simulation under the virtual topology network using ns-2, the network simulation tool are presented.*

Key words : Network Security, Trace-back, Hacking, Simulation, Information Security

1. 서 론

컴퓨터, 네트워킹 기술의 발전과 인터넷 문화의 보급으로 인해 지식 정보화 사회로의 이전이 급속화 되고 있다. 그러나 인터넷은 개방적인 특성과 네트워크 프로토콜의 근본적인 문제점 및 정보 시스템 자체의 취약성 등으로 인해서 악의적인 불법 침입에 의한 접근 정보의 오용 및 도청, 위조 및 변조 행위들이 쉽게 이루어 질 수 있다. 따라서 인터넷을 이용한 각종 침해 사고가 급증함에 따라 능동적인 해킹방지 기술이 절실히 요구되고 있으나, 현재까지 해킹방지를 위해 사용한 각종 시스템들은 단순히 해커의 해킹 성공률을 낮추기 위한 방법이었고, 해킹 시도 자체를 제한하지는 못하고 있다. 그래서 이에 대한 능동적인 대응방법으로 해킹을 시도하는 해커의 실제 위치를 실시간으로 추적하는 침입자 역추적 기술에 대한 필요성이 점차 커지고 있다.

침입자 역추적 기술은 일반적으로 해커가 우회공격을 시도하는 경우, 해커의 실제 위치를 추적하는 기술과, IP주소가 변경된 패킷의 실제 송신지를 추적하는 2가지 기술로 분류된다. 이때, 우회 공격을 시도하는 해커의 실제 위치를 추적하는 기술을 TCP 연결 역추적(TCP connection trace-back)이라 하고, IP 주소가 변경된 패킷의 실제 송신지를 추적하는 기술을 IP 패킷 역추적

† 종신회원 : 한국전자통신연구원 네트워크보안구조연구팀 팀장
bluesea@etri.re.kr
†† 정 회 원 : 한국전자통신연구원 네트워크보안연구부 연구원
rinyfeel@etri.re.kr
††† 종신회원 : 충북대학교 전기전자컴퓨터공학부 교수
shlee@chungbuk.ac.kr
논문접수 : 2004년 3월 12일
심사완료 : 2004년 11월 4일

(IP packet trace-back)이라 한다[1].

본 논문에서는 패킷 워터마크 기법을 이용하여 실시간 역추적이 가능한 TCP 연결 역추적 시스템인 RTS (network-based Real-time connection Trace-back System)를 제안하고, 다양한 IP 네트워크를 시뮬레이션 할 수 있는 UC Berkeley에서 개발한 분산 이벤트 구동 네트워크 시뮬레이터인 ns-2 도구를 사용하여 제안된 시스템을 모델링하고 시뮬레이션을 통한 알고리즘 검증을 수행 하였다. 따라서, 본 논문의 구성은, 2장에서는 기존의 역추적 기술에 대해 살펴보고, 3장에서는 본 논문에서 제안하는 TCP 연결 역추적 시스템인 RTS의 알고리즘 특징과 동작 방식을 기술하며, 4장에서는 RTS 역추적 알고리즘에 대해 ns-2 시뮬레이터를 기반으로 한 역추적 구성요소를 모델링 하여 가상 네트워크 토폴로지 환경에서 역추적 시뮬레이션 실험 내용 및 실험 결과를 분석한 후, 마지막으로 결론 및 향후 연구 과제를 제시한다.

2. 기존의 역추적 기술

침입자 역추적 기술이란 해킹을 시도하는 해커의 실제 위치를 실시간으로 추적하는 기술을 말한다. 역추적 기술은 일반적으로 해커가 우회공격을 시도하는 경우, 해커의 실제 위치를 추적하는 기술과, IP주소가 변경된 패킷의 실제 송신지를 추적하는 2가지 기술로 분류된다. 이때, 우회 공격을 시도하는 해커의 실제 위치를 추적하는 기술을 TCP 연결 역추적이라 하고, IP 주소가 변경된 패킷의 실제 송신지를 추적하는 기술을 IP 패킷 역추적이라 한다.

TCP 연결 역추적 기술은 크게 2가지로 분류할 수 있다. 이는 호스트 기반 연결 역추적(host-based connection trace-back) 기술과 네트워크 기반 연결 역추적 (network-based connection trace-back) 기술로 분류된다. 본 장에서는 TCP 연결 역추적 알고리즘을 중심으로 기술한다[2].

2.1 호스트 기반 연결 역추적 시스템

호스트 기반 연결 역추적 기술은 역추적을 위한 모듈이 인터넷상의 호스트들에 설치되는 역추적 기법으로 호스트에서 발생하는 로그 기록 등의 다양한 정보를 바탕으로 역추적을 진행하는 기술이다. 그러나 이러한 방법을 이용하여 역추적을 수행하기 위해서는 인터넷상의 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. 다음 표 1은 기 제안된 호스트 기반 역추적 기술을 정리하였다.

2.2 네트워크 기반 연결 역추적 시스템

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.

3. 제안 알고리즘

본 논문에서 제안하는 RTS는 기존의 역추적 시스템들과는 달리 비록 모든 중간 경유지를 확인할 수 없다 하더라도 해커의 최종 위치를 찾을 수 있는 가능성이 존재하는 역추적 시스템이다. RTS에서 사용하는 패킷 워터마크 기법은 SWT에서 사용한 방식과 동일한 방법을 이용한다. 그러나, 전체 시스템을 비교해 볼 때 RTS는 현재 인터넷 환경에 적용이 가능하도록, 이미 구축된 네트워크 장비들의 변경을 최소화하도록 설계했다는 점

표 1 호스트 기반 역추적 기술

종류	내용
CIS (Caller Identification System)	사용자가 특정 시스템에 접속할 때, 해당 시스템은 접속을 시도하는 사용자가 거쳐 온 모든 시스템에 대한 시스템 목록과 로그인 ID등의 정보를 이용하여 역추적에 활용 하는 시스템으로 접속을 원하는 사용자가 거쳐 온 시스템 각각에 대한 인증을 거쳐 가는 시스템마다 요구하므로 이로 인한 네트워크 부하가 크고, CIS에 오고 가는 인증을 위한 메시지의 무결성을 보장하지 못하는 단점을 가진다[3].
AIAA (Autonomous Intrusion Analysis Agent)	침해를 당한 서버의 해킹 피해 분석과 추적을 위한 로그 분석을 에이전트를 이용해 자동화한 역추적 시스템으로 역추적 경로 상에 존재하는 시스템들의 관리자 도움을 받아 에이전트를 설치하기 때문에 역추적을 완료하기까지 많은 시간이 필요하고 모든 시스템에 직접 접속해야 하기 때문에 관리자와의 협조가 불가능하여 시스템으로의 접근이 불가능한 경우 역추적 자체가 불가능하다[4].

표 2 네트워크 기반 연결 역추적 기술

종류	내용
Thumbprints 기반 알고리즘	본 알고리즘은 공격자의 시스템으로부터 해커가 위치하고 있는 시스템까지의 연결 사슬을 통해 송수신되는 데이타는 동일할 것이라는 가정 하에 송수신 되는 데이타로부터 추출한 내용을 특정 함수의 입력으로 사용하고 이를 통해 얻은 정보(thumbprints)가 일정 수준이상 동일한 경우 두 연결은 하나의 연결 사슬 상에 존재하는 것으로 판단한다. 그러나 패킷이 암호화되어 있는 경우, 연결 사슬 구성이 불가능하며, False Positive와 False Negative가 발생할 수 있다[5].
Timing 기반 알고리즘	해커가 입력하는 키보드 입력에 의해 발생하는 데이타 송신 간격은 프로그램이 송신하는 데이타에 비해 매우 크기 때문에, 이를 쉽게 파악할 수 있고, 만약 같은 연결 체인에 속한다면 그 간격이 매우 유사할 것이라는 점을 이용하는 시스템이다[6].
TCP 시퀀스 넘버의 증가 정도를 이용한 알고리즘	TCP 시퀀스 넘버를 이용하는 알고리즘은 송수신되는 데이타가 암호화 되더라도 데이타의 양은 크게 변하지 않는 다는 점에 착안하여 시퀀스 넘버의 증가 정도를 변동 폭의 조정을 통해 비교하고 연결 체인을 구성하는 알고리즘이다[7].
Sleepy Watermark Tracing(SWT)	SWT 역추적 기법은 가장 최근에 발표된 것으로, 기존 역추적 시스템들의 문제점을 상당부분 개선하였다. SWT는 가디언 게이트웨이(guardian gateway)와 가디언 호스트(guardian host)로 구성되며, 공격이 발생하면 역추적을 위해 가디언 호스트의 Watermark Enabled Application 이 해커의 공격에 대한 응답 패킷에 워터마크를 삽입한다. 그리고, 역추적이 시작되면 가디언 게이트웨이의 Active Tracing 모듈과 연동되어 워터마크가 삽입된 패킷을 찾기 시작한다. SWT는 공격을 당한 네트워크 망뿐만 아니라 인터넷과 연결된 모든 네트워크 망에 Watermark Enabled Application을 포함하는 가디언 호스트 및 가디언 게이트웨이의 설치가 요구된다. 이는, 현재 인터넷 망에 구축되어 있는 네트워크 장비 및 경로의 변경이 요구됨으로 현실적으로 적용하기가 어렵다[8].

에서 SWT와 차별을 가진다. 즉, RTS는 첫째, 다양한 기존 보안 시스템과 연동할 수 있는 중립적인 구조를 가지고 있다. 예를 들어, 침입탐지 정보 획득의 중립성과 패킷 차단 기능 수행의 중립성이다. 이는 다시 말해 기존의 어떠한 장비도 사용할 수 있음을 뜻한다. 둘째, RTS의 각 블록간의 독립성을 통한 유연한 구조를 채택하고 있다. 따라서 구현자의 의도에 의해 각 블록의 설계를 독립적으로 수행할 수 있다.

본 장에서는 패킷 워터마크 기법, RTS의 특징과 동작 방식에 대해 기술한다[9].

3.1 알고리즘 특징

3.1.1 패킷 워터마크 기법

RTS에서 패킷 워터마크는 특정 패킷을 식별 가능하도록 만들기 위해 의미적인 워터마크 정보를 사용하여 패킷의 이동 경로에 대한 역추적을 용이하게 하는 기법으로 사용한다. 여기에서 사용되는 워터마크 정보는 직관적으로 식별 불가능한 디지털 워터마크와는 달리 패킷 캡쳐를 통한 분석에서 쉽게 식별되는 정보다. 그러나 응용 프로그램의 사용자인 수신자에게는 보이지 않는 의미적인 워터마크 정보이다. 의미적인 패킷 워터마크의 구조는 그림 1과 같이 패킷 워터마크의 시작임을 나타내는 워터마크 판단 신호인 'WM', 워터마크를 생성한 호스트의 IP 주소, 그리고 워터마크 식별자 또는 공격자 식별 필드, 백스페이스 문자로 이루어져 있다. 여기에서 백스페이스 문자는 수신되는 곳에서 워터마크 정보를 지우도록 함으로써 수신자에게 워터마크 정보가 들어나지 않도록 하는 역할을 한다.

패킷 워터마크 기법에 의해 삽입된 패킷이 RTS에 의

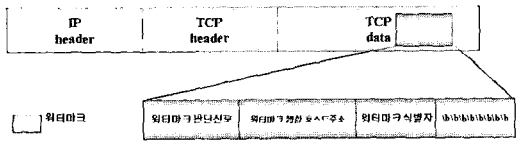

그림 1 워터마크 패킷 구조

해 검출이 되면 해당 워터마크를 추출하고 추출된 워터마크로부터 얻은 정보를 이용하여, 최초 워터마크를 삽입한 RTS로 워터마크 검출 결과를 전송한다. 이때, 워터마크 유무는 워터마크 정보의 최초 부분인 워터마크 판단 신호를 보고 판단한다. 일단 워터마크가 존재한다는 것을 판단하면, 전체 워터마크 정보를 추출하고 워터마크 생성 호스트 주소 부분을 통해서 최초 워터마크 생성 호스트의 정확한 IP 정보를 추출한다.

RTS 알고리즘에서 사용되는 패킷 워터마크 기법과 유사하게 패킷 마킹을 응용한 기존의 역추적 알고리즘들은 패킷이 이동하는 경로의 라우터에서 패킷의 헤더에 경로 정보를 삽입함으로써, 실제적으로 공격이 발생했을 때 역추적을 수행하기 보다는 공격을 수행하는 패킷의 헤더에 의존하여 해커의 정보를 유추한다. 또한 기존의 패킷 마킹 기술들을 사용하기 위해서는 패킷이 거치는 라우터의 기능이 경로 정보를 삽입할 수 있도록 수정되어야 하며 패킷을 라우팅 하는 프로토콜에 대한 변경이 요구된다. 그러므로, 기존의 기술을 사용하여 해커의 실제적인 위치에 대한 역추적을 수행하기 위해서는 패킷 마킹을 지원하는 라우터를 네트워크의 경계에 설치하여야 하고 패킷 라우팅과 더불어 패킷의 헤더에 마킹을 수행하는 새로운 프로토콜을 요구하기 때문에

현재의 인터넷 환경에 적용하는 것은 불가능하다. 그리고 기존의 패킷 마킹 기술들은 패킷의 내용보다는 패킷의 헤더에 마킹을 삽입함으로써 여러 호스트를 경유하는 해커의 공격에 대한 역추적은 사실상 불가능하고 해커에 의해 이용당한 최종 경유 호스트를 공격자로 오인시키는 문제점을 가진다. RTS 알고리즘은 이러한 문제점을 해결하여 해커에 의해 해킹 당한 피해 시스템의 응답 패킷의 내용에 식별 가능한 의미적인 패킷 워터마크를 삽입하여 패킷의 역추적을 용이하게 하고 우회 경로 공격을 수행하는 해커의 실제적인 위치를 실시간으로 찾을 수 있는 기법이다.

본 논문에서 제안하고 시뮬레이션을 통해 검증하고자 하는 RTS의 특징은 다음 표 3과 같다.

3.2 시스템 구성 및 동작 시나리오

제안하는 RTS 시스템은 그림 2와 같이 침입 탐지 시스템, 패킷 차단 시스템, 경로 역추적 시스템으로 구성되며, 다음과 같은 제한 사항을 갖는다.

(1) 해커에 의해 시도되는 해킹은 TCP 연결을 유지하는 해킹 기법이라 가정하고,

(2) 일정 수 이상의 네트워크에 RTS 시스템이 설치되었다고 가정하며,

(3) TCP 연결은 암호화되지 않았다고 가정한다.

- 침입 탐지 시스템 : 침입을 탐지하여 역추적을 시작하는 역할을 수행하는 침입 탐지 시스템은 기존에 널리 사용되고 있는 일반 네트워크 기반 침입 탐지 시스템을 사용할 수 있다. 다만 경로 역추적 시스템과의 연동이 가능하면 된다.
- 패킷 차단 시스템 : 패킷 차단 시스템은 패킷 차단 모듈과 워터마크 생성 모듈로 구성되어 있는데, 패킷 차단 모듈은 경로 역추적 시스템의 요청에 따라 해커의 해킹에 대한 피해 시스템의 응답 패킷을 출발지 및 도착지 IP주소와 출발지 및 도착지 port 번호를 이용하여 차단하는 기능을 수행한다. 그리고, 워터마크 생성 모듈은 응답 패킷에 삽입할 워터마크를 생성하고, 이를 공격에 대한 응답 패킷에 삽입하여 외부로 송신하는 기능을 수행한다.
- 경로 역추적 시스템 : 경로 역추적 시스템은 워터마크 탐지 모듈, 경로 추적 모듈로 구성되어 있는데, 침입 탐지시스템의 침입 경보에 따라 패킷 차단 시스템에게 특정 패킷을 차단할 것을 지시하고, 그리고 송수신되는 패킷을 지속적으로 감시하여 워터마크가 삽입된 패킷의 존재여부를 판단하며, 워터마크가 삽입된 패킷이 감지되는 경우, 해당 패킷을 송신한 RTS 시스템으로 감지 결과를 전송한다.

RTS 시스템은 다음 그림 3과 같이 동작한다.

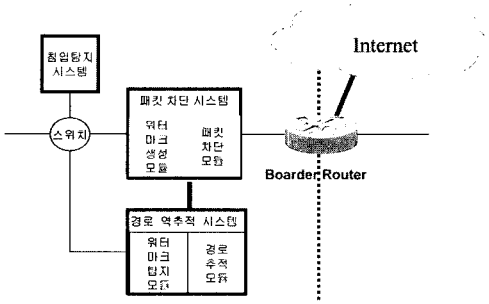

그림 2 RTS 시스템 구조

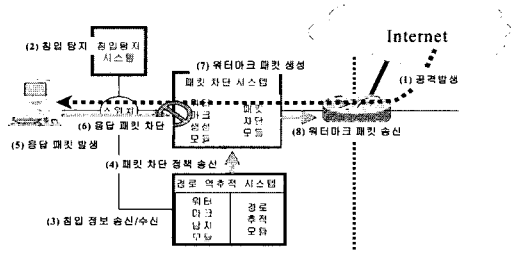

그림 3 내부 네트워크에서의 RTS 동작 방식

표 3 RTS의 특징

특징	내 용
실시간 역추적 가능	RTS는 저장된 로그 기록의 분석을 통해 역추적을 수행하는 것이 아니라, 해커의 공격에 대한 응답 패킷에 특정 마크를 삽입하고 이를 탐지함으로써 역추적을 수행하기 때문에, 실제 역추적에 소요되는 시간은 피해 시스템에서 실제 해커의 위치까지의 패킷 전달 시간과 추가적인 상수 시간뿐이다. 따라서 기존의 역추적 방식에 비해 매우 빠르게 역추적이 가능하다.
높은 역추적 성공률이 가능	현재 제안되어 있는 역추적 시스템들은 대부분 단계적인 역추적 기법을 사용하기 때문에, 역추적 경로 상의 중간 경유지 중 하나의 시스템이라도 역추적을 위한 정보를 가지고 있지 않는 경우, 역추적 자체가 불가능하게 되어 실제 해커의 위치를 찾을 수 없다. 그러나 RTS 시스템은 비록 몇몇 경유 시스템을 파악하지 못한다 하더라도 역추적에 성공할 수 있는 가능성을 가지고 있다.
워터마크 탐지 성공률 개선 가능	워터마크 탐지 모듈을 하나의 프로그램으로 개발하여 워터마크 탐지 모듈만을 따로 ISP의 백본 라우터에 설치할 수 있게 되면 역추적 경로 구성 및 해커의 위치 파악 성공률을 크게 증가시킬 수 있다.
암호화된 패킷 역추적 성공 가능성 존재	RTS 시스템은 워터마크를 패킷의 데이타 부분에 삽입하기 때문에 패킷이 암호화 되는 경우에는 역추적이 불가능할 수도 있다. 그러나, 역추적 경로상의 전체 연결이 암호화된 것이 아니라면 역추적에 성공할 수 있는 여지를 갖게 된다.

(1)공격자에 의해 침입이 시도된다. (2)침입탐지시스템에 의해 침입이 탐지된다. (3)침입탐지시스템에 의해 경로 역추적 시스템으로 경보가 송신되고 이를 경로 역추적 시스템은 침입 발생 정보를 수신한다. (4)경로 역추적 시스템이 침입과 연관된 연결에 대한 정보를 이용하여 침입차단시스템에 적용할 정책 작성하고 이를 침입차단시스템으로 송신한다. (5)공격자의 공격에 따른 피해 시스템의 응답이 송신된다. (6)경로 역추적 시스템에서 작성된 정책에 따라 패킷을 차단한다. (7)워터마크가 생성되고 수집된 패킷에 삽입된다. (8)워터마크가 삽입된 패킷을 송신한다.

일단, 워터마크가 삽입된 패킷이 공격 시스템으로 송신되면 그림 4와 같은 시나리오가 적용된다.

(9)워터마크가 삽입된 패킷송신을 통해 역추적이 시작된다. (10)워터마크가 삽입된 패킷이 탐지되고, (11)워터마크 발견 신호가 전송된다. (12)워터마크가 삽입된 패킷이 탐지 2번째로 탐지되고, (13)워터마크 발견 신호 역시 전송 추가적으로 전송된다. (14)워터마크가 삽입된 패킷이 해커가 존재하는 네트워크에서 탐지되어, (15)워터마크 발견 신호가 전송되면, (16)최초 역추적을 시작한 RTS는 워터마크 패킷 탐지 정보를 이용하여 역추적 경로를 구성하고 해커의 위치를 파악하게 된다.

4. 시뮬레이션을 통한 검증

네트워크 시뮬레이터인 ns-2를 이용한 기존의 네트워크 성능 모델링은 지연(delay), 네트워크 구성 등의 네트워크 망의 성능분석을 주 목적으로 모델링 된다. 반면에 역추적의 모델링은 여러 경유지를 거쳐 TCP 연결 체인을 이용한 공격과 이에 대한 역추적의 모델링이 되어야 하므로 다른 관점의 접근이 요구된다. 따라서, 다음과 같은 워터마크 삽입 기법을 이용한 역추적 시스템의 특징을 반영하여야 한다.

첫째, 역추적을 수행하는 에이전트는 공격에 대한 응답 패킷을 차단하고, 새로운 응답 패킷을 생성하여 워터마크를 삽입할 수 있어야 한다.

둘째, 역추적 에이전트는 노드를 지나가는 워터마크 패킷들의 탐지가 가능하여야 하며, 최초 워터마크를 생성한 역추적 노드에 탐지된 워터마크 정보를 송신할 수 있어야 한다.

셋째, 하나의 노드를 대상으로 시도되는 공격은 최소한 한 개 이상의 노드를 경유하여 공격하는 TCP 기반 연결 체인 공격이어야 한다.

본 장에서는 상기 특징을 반영한 역추적 구성요소 모델링 방법을 기술하였으며, 모델링 되어진 역추적 구성요소가 포함된 가상 네트워크 토폴로지 환경에서 역추적 시뮬레이션 실험 내용과 결과를 분석한다.

4.1 역추적 구성요소 모델링

역추적 구성 요소 모델링에서는 ns의 네트워크 기본 구성 요소인 가상 패킷 구조, 에이전트, 어플리케이션을 본 논문에서 제안하는 RTS 역추적 모델링에 맞게 패킷의 유형을 새롭게 정의한 패킷, 역추적 기능을 수행하는 역추적 에이전트, 다수의 경유지를 거친 TCP 연결 체인 공격 발생 및 데이타 처리 동작 특징을 반영한 공격처리 어플리케이션으로 새롭게 정의하였다.

4.1.1 역추적 에이전트(RTS agent)

워터마크 삽입 기법을 이용한 RTS 역추적 동작 특성을 모델링한 역추적 에이전트의 동작상태 전이도는 그림 5와 같으며, 총 8개의 동작 상태가 있다.

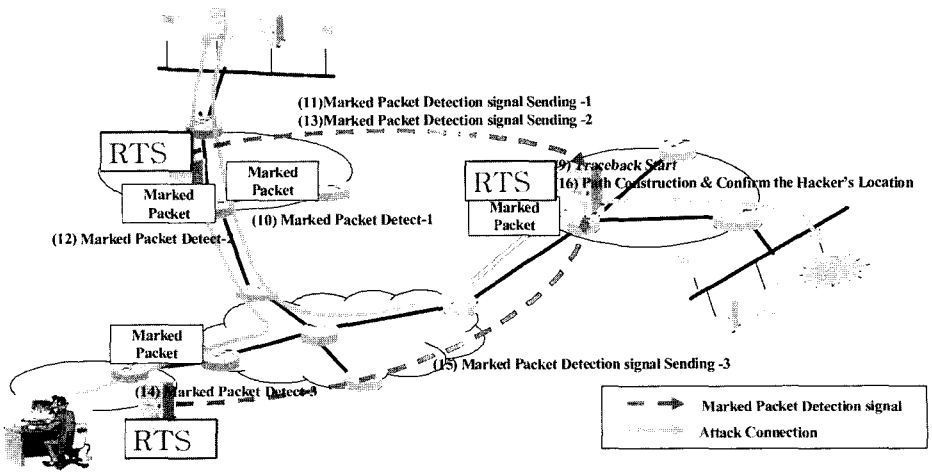

그림 4 외부 네트워크에서의 역추적

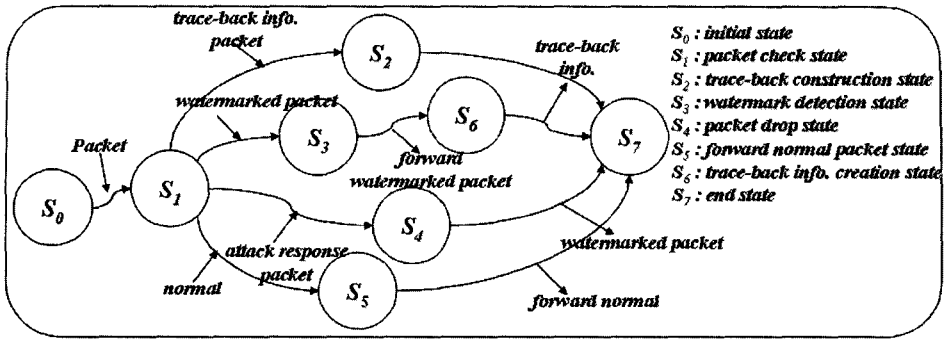

그림 5 역추적 에이전트의 동작상태 전이도

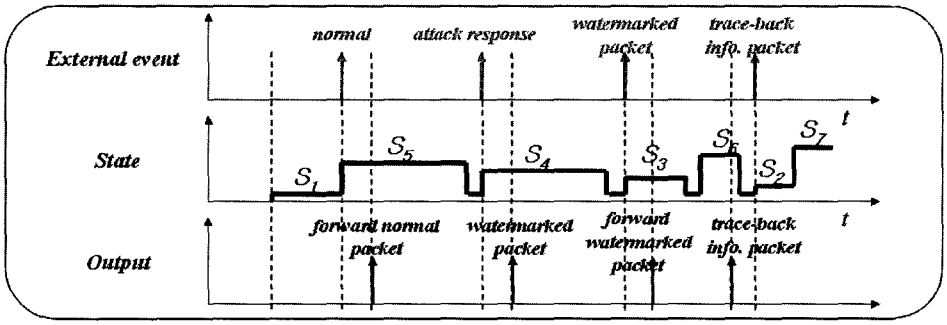

그림 6 이벤트 발생상태 타이밍 전이도

그림 6은 역추적 에이전트 동작상태 전이도에서 외부 이벤트(external event), 상태(state), 출력 이벤트(output)로 구성된 이벤트 발생상태 타이밍 전이도이다. 외부 이벤트로 정상(normal), 공격 응답(attack response), 워터마크 삽입 패킷(watermarked packet), 역추적 정보 패킷(trace-back info. packet) 이벤트가 입력으로 발생할 때마다 역추적 에이전트 동작상태 전이도에서 처리되는 상태 전이 과정과 해당 상태에서 출력되는 출력 이벤트(output)를 도식화한 것이다.

다음은 역추적 에이전트 내에서 최종 공격의 응답 패킷에 워터마크를 삽입한 패킷에 대한 워터마크 탐지 동작과 이에 따른 역추적 경로 설정과 노드 간 연결 재설정에 관한 동작들을 모델링 하였다.

네트워크 i는 최소 1개의 역추적 노드(n_{i-1})와 최소 1개 이상의 일반 노드(a_{i-1})로 구성 되어 있다. 역추적 노드는 역추적 에이전트가 연결된 노드를 말하며, 일반 노드는 공격자(attacker), 희생자(victim), 중간경유(via), 일반 노드가 될 수 있는 노드를 말한다. 또한, 네트워크 1 은 공격자가 포함된 네트워크이고, 네트워크 k는 공격대상 노드가 포함된 네트워크이다. 그리고, 경유 노드가 $a_{i-1}...a_{k-1}$ 라 할 때, TCP 연결 공격은 a_0, a_{i-1}, a_i, a_{k-1} 순으로 진행한다고 가정한다.

역추적 패킷 헤더에 따라, 구성된 i 번째 RTS 패킷 헤더는 다음과 같다.

HEADER[i]=>(RL addr.[i], Packet Type[i], logical addr.[i], trace path list[i], attack path list[i])

실 주소 쌍(RL addr.:real link address)은 노드 간의 패킷을 전송하기 위한 실제 소스 어드레스(SA:source address), 대상 어드레스(DA:destination address) 쌍을 나타내는 필드이다. 패킷 타입(Packet type)은 워터마크 삽입 패킷, 워터마크 탐지 패킷의 유형을 구분하기 위한 필드이다. 공격 경로 리스트(attack path list[i])는 i 번째 연결체인 공격을 시도한 노드 간의 SA, DA 주소 쌍이다.

논리적 주소 쌍(logical addr.[i])는 연결체인 공격에 이용된 노드 간의 SA, DA 쌍, 즉 공격 경로 리스트(attack path list[j])의 역 주소 값이다. 역추적 경로 리스트(trace path list[])는 워터마크를 탐지한 역추적 노드의 주소를 포함하기 위한 주소 리스트 필드이다. 이러한 네트워크 구조와 역추적 패킷 구조를 기반으로 역추적 에이전트에서 워터마크가 생성되고 삽입된 패킷이 최종 공격자까지 패킷이 흘러가는 워터마크 응답 패킷 전송 흐름과 워터마크 패킷이 역추적 에이전트에서 탐지되어 최초 워터마크 패킷을 생성한 역추적 에이전트

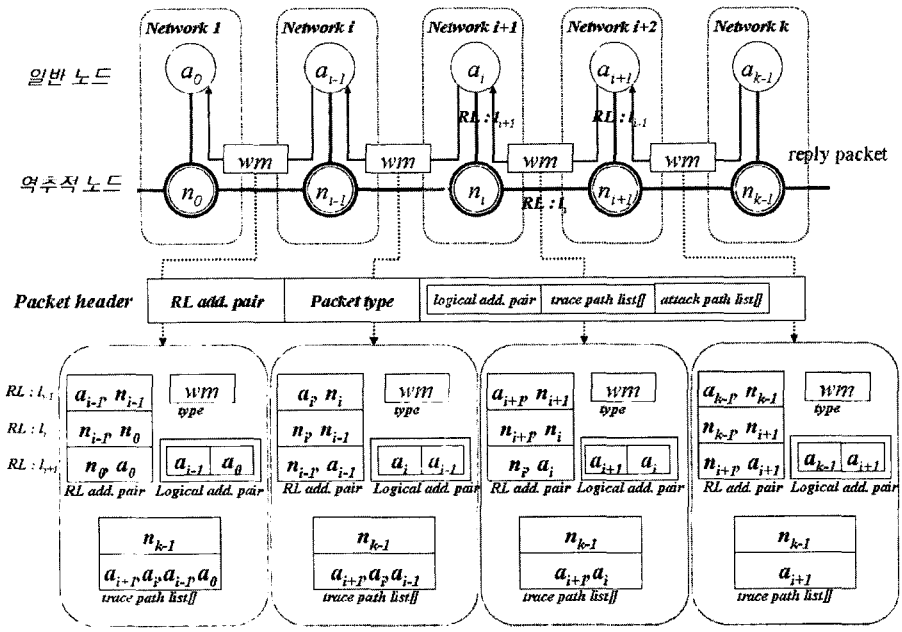

그림 7 워터마크 응답 패킷 전송 흐름

로 탐지 정보가 흘러가는 워터마크 탐지 흐름 두 가지로 나누어 살펴본다.

가. 워터마크 응답 패킷 전송 흐름

역추적 에이전트에서 워터마크가 생성되고 삽입된 패킷이 최종 공격자까지 패킷이 흘러가는 워터마크 응답 패킷을 전송하기 위한 워터마크 응답 패킷 전송 흐름 동작 모델링은 그림 7과 같다.

attack_path_list[i] 에서 i번째 attack_path[i] 쌍은 (a_i, a_{i+1})이다. 따라서, logical_path[i]는 attack_path[i]의 역 주소 쌍이므로, 노드 a_{i+1}에서 a_i까지 워터마크가 포함된 워터마크 응답 패킷은 a_{i+1}, n_{i+1}, n_i, a_i 4개의 노드를 거친다. 이때, 4개의 노드에 연결되는 링크는 l_{i+1}, l_i, l_{i-1}가 된다. 그러나, 패킷의 논리적인 레벨에서 SA, DA 쌍은 (a_{i+1}, a_i)가 되어야 하지만 노드의 물리적인 연결은 3개의 링크(l_{i+1}, l_i, l_{i-1})를 거치게 되므로, 3개의 RL 주소 쌍이 필요하게 된다. 따라서, 패킷 헤더가 링크(l_{i+1}, l_i, l_{i-1})를 지날 때 패킷 헤더의 값은 그림 7의 워터마크 응답 패킷 전송 흐름의 패킷 헤더 구조처럼 변한다.

나. 워터마크 탐지 패킷 이동 흐름

역추적 노드에서 워터마크를 탐지하고 탐지된 워터마크 정보를 최초 워터마크를 생성한 역추적 노드에 전송하기 위한 동작 모델링은 그림 8과 같으며, 패킷 구조는 앞에서 설명한 워터마크 응답 패킷 전송 구조와 같다. 이때, RL add. 주소 쌍의 SA는 워터마크를 탐지한 주소의 역추적 노드 주소와 최초 워터마크를 생성한 역추적 노드의 주소인 trace path list[0]의 쌍으로 이루어진다.

RL add. pair[i] = act_connect_path[i] = pair(current node(ni), trace path list[0])

또한, 패킷 내 역추적 경로 리스트(trace path list[]) 필드 값을 논리적 주소 쌍의 DA 주소를 추출하여 리스트에 추가 시킨다. 그리고, 워터마크 필드(wm) 값을 워터마크가 탐지되었다는 정보로 변경한 후에 설정된 경로에 따라 최초 워터마크를 생성한 역추적 노드로 전송한다.

4.1.2 역추적 공격 처리 어플리케이션(attack process application)

역추적 시뮬레이션을 수행하기 위해 어플리케이션은 하나의 노드를 대상으로 시도되는 공격은 최소한 한 개 이상의 노드를 경유하여 공격하는 TCP 기반 연결 체인 공격이어야 한다는 공격 특징을 반영하여야 한다. 따라서, 공격 처리 어플리케이션은 다수의 경유지를 거친 TCP 연결 체인 공격 발생과 이러한 데이타를 처리하기 위한 공격 동작 특징을 반영하여 모델링 하였다.

그림 9는 TCP 연결 체인 공격 경로 유형을 나타낸다. 전체 노드의 개수는 L개이고 공격자 노드(attacker node), 경유지 노드(via node), 희생자 노드(victim node)를 포함한 공격 대상의 노드 개수가 K개라 가정

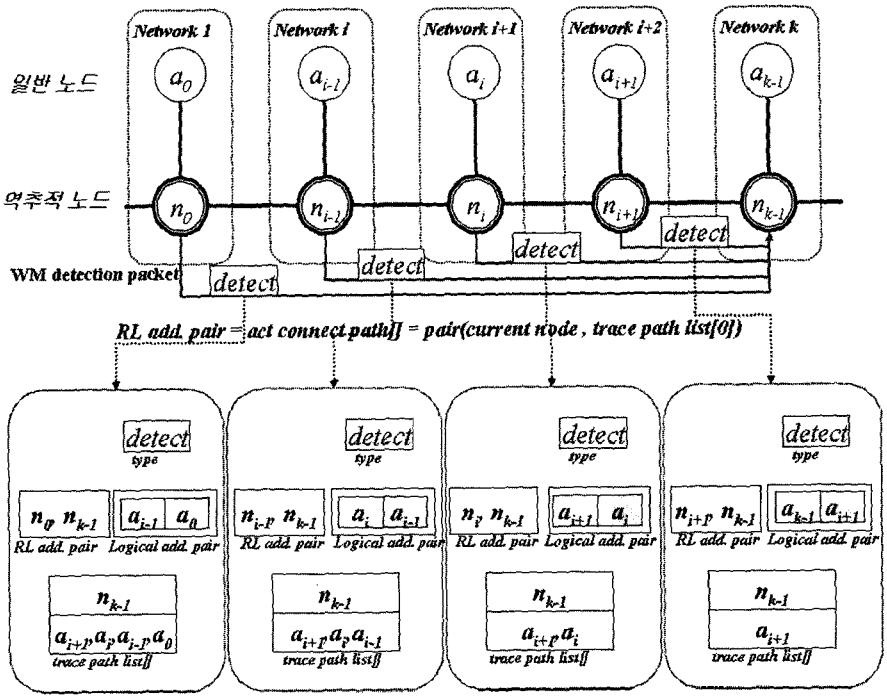

그림 8 워터마크 탐지 패킷 이동 흐름

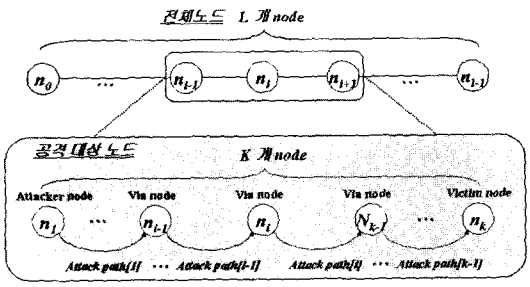

그림 9 공격 경로 유형

한다면, 총 공격 경로의 개수는 k-1개가 된다. 이때, k-1개의 공격 경로를 유형별로 분류하면 다음과 같다.

- 1번째 공격 경로 : attack path[1] = n[1], n[2] : 중간 공격 경로(via attack path)
- 2번째 공격 경로 : attack path[2] = n[2], n[3] : 중간 공격 경로(via attack path)

.....

- i번째 공격 경로 : attack path[i] = n[i], n[i+1] : 중간 공격 경로(via attack path)

.....

- k-2번째 공격 경로 : attack path[k-2] = n[k-2], n[k-1] : 기반 공격 경로(base attack path)
- k-1번째 공격 경로 : attack path[k-1] = n[k-1], n[k] : 최종 공격 경로(final attack path)

공격 경로는 중간 공격 경로, 기반 공격 경로, 최종 공격 경로 3가지 유형으로 구분한다. 최종 공격 경로는 최종 희생자 노드를 공격하는 k-1 번째 공격 경로인 attack path[k-1]이며, 기반 공격 경로는 희생자 노드를 최종 공격하기 위한 이전 노드를 공격하는 k-2 번째 공격 경로 attack path[k-2]이다. 그리고, 중간 공격 경로인 attack path[1]~attack path[k-3]은 공격자에서 시작하여 attack path[k-2] 이전까지의 경유 공격 패스이다.

4.2 역추적 시뮬레이션 실험 및 분석

본 절에서는 제안하는 RTS 동작을 네트워크 시뮬레이터 도구인 ns-2를 이용하여 가상 네트워크 토폴로지를 구성하고, 구성된 네트워크 토폴로지를 대상으로 TCP 연결 체인 공격을 시도하여 역추적 동작 시뮬레이션을 실험하고 실험 결과를 분석한다.

4.2.1 실험 환경

역추적 시뮬레이션을 위해 가상 네트워크 토폴로지는 그림 10과 같다. 가상 네트워크 토폴로지는 총 4개의 네트워크로 구성되어 있으며, 네트워크 1은 공격자 노드가 포함된 네트워크, 네트워크 2~3은 중간 경유지 노드가 포함된 네트워크, 네트워크 4는 최종 공격 대상이 포함된 네트워크이다. 각 네트워크는 하나의 라우터 노드와

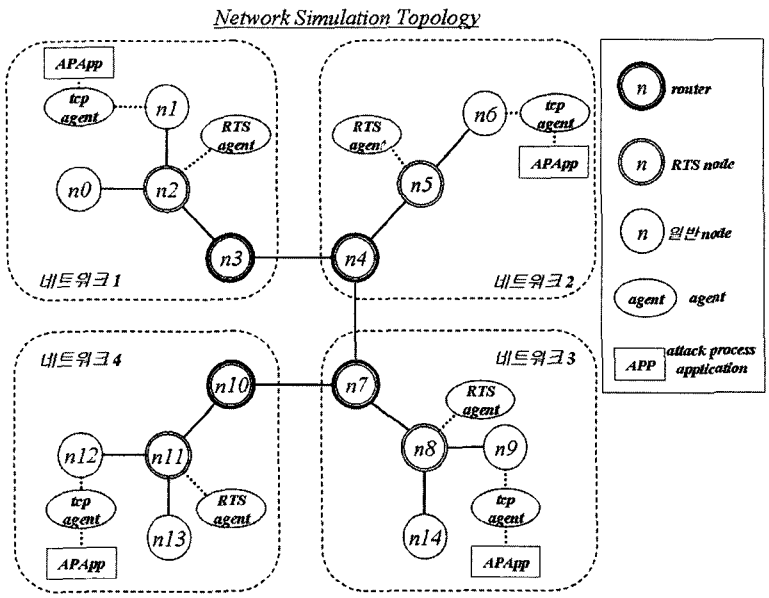

그림 10 가상 네트워크 시뮬레이션 토폴로지

역추적 노드, 한 개 또는 두 개의 일반 노드로 구성되어 있으며, 역추적 노드에는 역추적 에이전트(RTS agent)가 연결되어 있다. 또한, 일반 노드에는 TCP, TCPSink 에이전트 쌍이 연결되어 있으며, 역추적 어플리케이션(APApp : attack process application)이 연결되어 있다. 각 노드는 유니캐스트 라우팅을 지원하며, 각 노드와 연결된 링크는 duplex 링크, 대역폭 1/10/ 100M, 큐 구조는 DropTail, 전송지연 40ms로 구성하였다. 또한, 공격자 노드로 n_1, 최종 공격 대상인 희생자 노드로 n_{12}, 중간 경유 노드로 n_6, n_9, 역추적 노드로 n_2, n_5, n_8, n_{11} 노드를 설정하였다.

4.2.2 시뮬레이션 테스트 및 결과 분석

앞서 설정한 가상 네트워크 토폴로지 실험 환경에서 수행되는 시나리오는 공격자 노드에서 중간 경유 노드를 거쳐 최종 희생자 노드에 TCP 연결체인 공격을 시도한다. 즉, $n_1 \rightarrow n_6 \rightarrow n_9 \rightarrow n_{12}$의 TCP 연결 공격 흐름을 갖는다. 그리고 희생자 노드가 포함된 네트워크의 역추적 노드에서 워터마크 패킷을 생성하고, 워터마크 패킷이 거쳐 가는 역추적 노드에서 워터마크 패킷을 탐지하여 탐지된 내용을 최초 워터마크를 생성한 노드에 전송하는 과정을 시뮬레이션 한다.

다음은 역추적 시뮬레이션 수행 후 GnuPlot을 이용하여 역추적 시뮬레이션 trace 분석을 그래프의 형태로 표현한 결과를 보여주고 있다. 이 그림에서 X축은 시뮬레이션 타임이고 Y축은 패킷을 송/수신 하는 노드 ID 나타낸다.

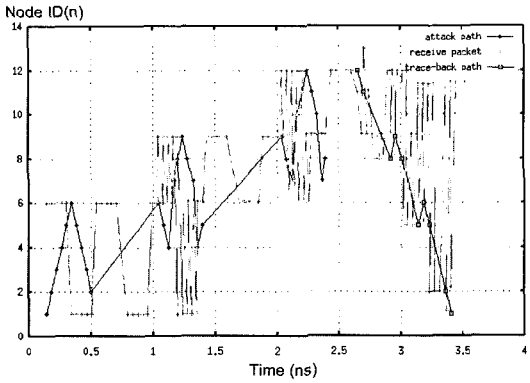

그림 11 시간 단위 별 각 노드에 도착하는 패킷의 이동 경로 그래프

그림 11은 시간 단위 별 각 노드에 도착하는 패킷의 이동 경로를 3가지 패턴으로 분석한 결과이다. 3가지 패턴은 공격 패킷의 이동 경로 흐름(attack path), 역추적을 위해 워터마크가 삽입된 응답 패킷의 이동 경로 흐름(trace-back path), 시뮬레이션 수행 동안 패킷 송/수신 노드(receive packet)의 정보를 보여주고 있다. 그래프의 공격 패킷의 이동 경로 흐름을 분석하면, 역추적 시뮬레이션 시나리오에 의해 TCP 연결 체인 공격은 노드 1→6→9→12 임을 알 수 있다. 그래프의 워터마크 삽입된 응답 패킷의 이동 경로 흐름(trace-back path)을 분석하면, 패킷의 이동 경로가 노드 12 → 11 → 8 → 9 → 8 → 5 → 6 → 5 → 2 → 1 순으로 흘러가는 것을

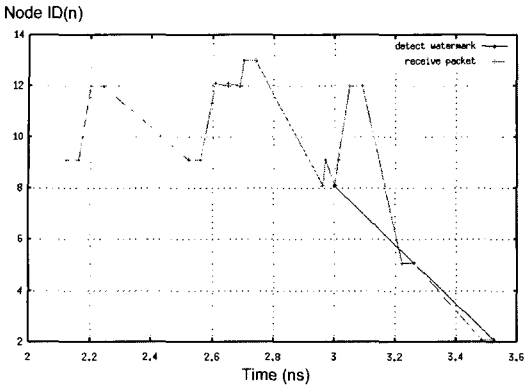

그림 12 최초 역추적 노드에서 수신되는 패킷의 발신 노드 정보

확인할 수가 있다. 따라서, 응답 패킷의 이동 경로 흐름(trace-back path) 그래프 곡선의 상위 꼭지점 노드 9, 6, 1과 공격 패킷의 이동 경로 흐름(attack path)의 경로 노드 1, 6, 9가 일치하여 TCP 연결 체인 공격을 시도한 TCP 연결의 역추적이 가능함을 보여주고 있다.

그림 12는 TCP 연결 체인 공격을 탐지하여 워터마크를 생성하는 최초 역추적 노드(노드 11)에서 수신되는 패킷의 발신 노드 정보를 2가지 패턴으로 분석한 결과이다. 수신 패킷(receive packet) 패턴은 노드 11에서 수신되는 패킷의 발신 노드 정보(receive packet)를 나타내며, 워터마크 탐지(detect watermark) 패턴은 워터마크를 탐지한 역추적 노드에서 발신한 노드 정보를 나타낸다. 워터마크 탐지(detect watermark) 패턴을 분석하면, 노드 11로 워터마크 탐지 정보를 노드 8, 5, 2의 순서로 송신하였음을 확인할 수 있다.

5. 결 론

본 논문에서는 기존 역추적 기술이 주로 사용해온 로그 분석 방법을 탈피하여 실시간으로 역추적을 수행할 수 있는 RTS라는 역추적 시스템을 제안하고, 네트워크 시뮬레이터 도구인 ns-2를 이용하여 TCP 연결체인 형태의 공격을 시도하는 공격자의 근원지를 추적하는 패킷 워터마크 기법을 이용한 RTS의 동작 검증을 수행하였다. RTS의 동작 검증은 역추적 기능을 수행하는 역추적 에이전트와 TCP 연결 체인 공격을 처리하는 공격처리 어플리케이션으로 이루어진 역추적 네트워크 구성요소를 기반으로 한 가상의 네트워크 토폴로지 환경을 구축하고 역추적 알고리즘 검증 실험을 하였다. 그리고, 실험을 통해 trace 분석 결과를 얻고, 분석된 결과를 통해 IP 네트워크 망에서 패킷 워터마크 기법을 이용한 역추적 시스템이 설치되었을 때, 역추적 기능 동작이 가능함을 시뮬레이션을 통해 확인하였다. 또한, 시뮬레이션을 통해 검증된 RTS를 통해 보다 원활한 역추적 알고리즘을 이용한 시스템 설치가 가능해져 해커의 해킹에 능동적으로 대응할 수 있는 역추적 시스템이 널리 활용 될 수 있을 것으로 생각된다.

마지막으로 본 논문에서는 네트워크 시뮬레이션 도구인 ns-2를 이용하여 RTS 알고리즘의 시뮬레이션을 통한 동작 검증을 포커스로 하였기에, 기존 알고리즘과의 성능 비교는 다루지 않았다. 따라서, 개념적으로 제안단계의 알고리즘인 SWT와 모델 수준의 관점에서 시뮬레이션을 통한 성능 비교와 F/W이나 IDS 기능을 수행하는 보안 노드의 상세 모델링 및 역추적 시스템이 설치된 다양한 네트워크 토폴로지 시뮬레이션 환경 하에서 역추적 패킷의 손실률 등의 성능 분석은 향후 연구과제로 남긴다.

참 고 문 헌

[1] 최양서, 서동일, 손승원, "역추적 기술 동향(TCP Connection traceback 중심)", ETRI 주간 기술 동향, 제1079호, pp13-25. 2003.

[2] 최양서, 김환국, 서동일, 이상호, "Connection Redirection 기법을 이용한 네트워크기반 실시간 연결 역추적 시스템의 설계", COMSW2003, pp.115-119, 2003.

[3] H. T. Jung et al. "Caller Identification System in the Internet Environment," Proceedings of the 4th Usenix Security Symposium, pp.69-73, 1993.

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

[5] S. Staniford-Chen and L. T. Heberlein. "Holding Intruders Accountable on the Internet," In Proceedings of the 1995 IEEE Symposium on Security and Privacy, pp.39-49, 1995.

[6] Y. Zhang and V. Paxson, "Detecting Stepping Stones," Proceedings of 9th USENIX Security Symposium, pp.171-184, 2000.

[7] K. Yoda and H. Etoh, "Finding a Connection Chain for Tracing Intruders," 6th European Symposium on Research in Computer Security-ESORICS 2000 LNCS -1985, pp.191-205, 2000.

[8] X. Wang, D. Reeves, S. F. Wu, and J. Yuill, "Sleepy Watermark Tracing: An Active Network-Based Intrusion Response Framework," Proceedings of IFIP Conference on Security, pp.369-384, 2001.

[9] Yangseo Choi, Dongil Seo, Seungwon Sohn, Sangho Lee, "Network-Based Real-Time Connection Traceback System(NRCTS) with Packet Marking Technology," In Proceedings of the 2003 International Conference on Computational Science and Its Applications, pp.31-40, 2003.

서 동 일

1989년 2월 경북대학교 전자공학과 졸업(공학사). 1994년 2월 포항공과대학교 정보통신공학과 졸업(공학석사). 2004년 8월 충북대학교 전자계산학과 졸업(이학박사). 1994년~현재 한국전자통신연구원 선임연구원. 2001년~현재 ASTAP Forum Information Security 의장. 관심분야는 인터넷 정보보호, 컴퓨터 통신, 네트워크 관리

김 환 국

1998년 2월 한국항공대학교 전자계산학과 이학사. 2000년 8월 한국항공대학교 컴퓨터공학과 공학석사. 2000년 9월~2002년 4월 이레스페이스 연구원. 2002년~현재 한국전자통신연구원 네트워크보안구조연구팀 연구원. 관심분야는 정보보호, 네트워크 시뮬레이션, 사이버테러 기술분석

이 상 호

1976년 2월 숭실대학교 전자계산 공학사. 1981년 2월 숭실대학교 대학원 시뮬레이션 공학석사. 1989년 2월 숭실대학교 대학원 컴퓨터네트워크 공학박사. 1981년~현재 충북대학교 전기전자 및 컴퓨터공학부 교수. 1999년~현재 한국정보과학회 충청지부 이사. 관심분야는 프로토콜 엔지니어링, 네트워크 보안, 네트워크 구조