계약망 프로토콜을 적용한 네트워크 보안 모델의 설계와 시뮬레이션*

서경진**, 조대호**

Application of Contract Net Protocol to the Design and Simulation of Network Security Model

Suh, Kyong Jin and Cho, Tae Ho

Abstract

With the growing usage of the networks, the world-wide Internet has become the main means to exchange data and carry out transactions. It has also become the main means to attack hosts. To solve the security problems which occur in the network such as Internet, we import software products of network security elements like an IDS (Intrusion Detection System) and a firewall. In this paper, we have designed and constructed the general simulation environment of network security model composed of multiple IDSes and a firewall which coordinate by CNP (Contract Net Protocol) for the effective detection of the intrusion. The CNP, the methodology for efficient integration of computer systems on heterogeneous environment such as distributed systems, is essentially a collection of agents, which cooperate to resolve a problem. Command console in the CNP is a manager who controls the execution of agents or a contractee, who performs intrusion detection. In the network security model, each model of simulation environment is hierarchically designed by DEVS (Discrete EVent system Specification) formalism. The purpose of this simulation is that the application of rete pattern-matching algorithm speeds up the inference cycle phases of the intrusion detection expert system and we evaluate the characteristics and performance of CNP architecture with rete pattern-matching algorithm.

Key Words: IDS, firewall, general simulation environment, network security model, CNP, DEVS formalism, rete pattern-matching algorithm

* 이 연구는 정보통신연구진흥원 (Institute of Information Technology Assessment)에 의해 지원되었음.

** 성균관대학교 정보통신공학부

1. 서론

네트워크 사용의 증가로 인터넷은 정보를 교환하고 거래를 수행하는 주요한 수단이 되었다. 반면에 인터넷은 네트워크에 존재하는 호스트를 공격하는 주요한 수단이 되기도 하였다 [1]. 인터넷과 같은 공중망에서 발생하는 보안상의 문제점을 해결하기 위해서 본 연구진은 침입 탐지 시스템 (IDS)이나 침입 차단 시스템 (firewall)과 같은 네트워크 보안 요소를 도입하였다. 침입 탐지 시스템은 컴퓨터나 네트워크 시스템의 불법적인 사용 (unauthorized use)이나 잘못된 사용 (misuse)을 인식하기 위해서 시스템 활동을 감시하는 것으로 네트워크 보안에 있어서 지극히 중요한 역할을 수행한다 [2,3]. 그리고 침입 차단 시스템은 인터넷과 내부 네트워크사이의 접속을 통제하는 수단이라고 할 수 있다 [4,5].

침입 탐지의 초기에는 외부로부터 보호하고자 하는 시스템마다 단일 호스트 기반의 침입 탐지 시스템 (host-based IDS)을 설치하여 대상 시스템마다 감시하고 보호하였는데 불필요한 자원의 낭비와 시스템 부하를 증가시키는 문제점을 발생시켰다. 또한 네트워크를 통해서 발생되는 침입을 탐지할 수 없었으며 각 시스템이 한정된 규칙만을 가지고 침입을 탐지하기 때문에 새로운 침입을 탐지하는데 많은 문제점이 발생되었다. 이와 같은 문제점들을 보완하고 탐지에 대한 성능을 향상시키기 위해서 네트워크를 기반으로 하는 다중 침입 탐지 시스템을 도입하게 되었다. 다중 침입 탐지 시스템은 분산 시스템의 이론을 적용한 다중 호스트 침입 탐지 시스템으로 네트워크에 분산된 에이전트들에게 작업을 분산시키므로 시스템 부하를 감소시켜 탐지의 속도를 향상시키고 발생된 침입에 알맞은 에이전트를 선택하게 하여 탐지의 성능을 향상시킬 수 있다. 무엇보다도 다중 침입 탐지 시스템에서는 보안 모델간의 연동이 시스템의 성능을 높이기 위한 중요한 요소이다. 따라서 에이전트에 근거한 연동을 설계하는데 있어서 높은 수행능력을 성취하기 위해서는 분산된 에이전트에게 효과적인 작업의 할당이 이루어져야 한다 [6]. 특별히 다중 에이전트의 연구에서 분산 인공지능 (Distributed Artificial Intelligence)의 새로운 기술에 대한 연구가 인공지능의 연구에서 중요한 분야가 되고 있으며 본 논문에서는 보안 시스템의 연동을 위해 계약망 프로토콜을 적용할 것이다 [7,8].

계약망 프로토콜은 분산된 에이전트들 중에서 bidding의 과정을 통해서 최상의 에이전트를 선택하고 선택된 에이전트는 서비스를 제공하게 된다 [9,10,11]. 이러한 작업의 분산을 통해 각 에이전트들은 효과적으로 주어진 작업을 처리한다. 또한 자신이 처리하지 못하는 작업의 경우는 다른 에이전트에게 의뢰하여 처리하므로 자신이 처리할 수 없는 작업의 처리도 가능하다. 이와 같은 특성을 침입 탐지에 적용하게 되면 신속하고 정확한 탐지가 가능하며 무엇보다도 새로운 침입을 탐지하는 능력을 향상시킬 수 있다.

시뮬레이션은 문제 해결의 대상이 되는 시스템이 시간에 따라 어떻게 변화하는지를 예측 또는 평가하는 것을 말하며 시스템을 축소 및 추상화한 모델을 통해 이루어지는데 실제 시스템에서 문제 해결을 하기에는 불가능하거나 위험한 일 또는 경비가 많이 드는 일들을 비교적 쉽게 처리할 수 있으므로 그 중요성이 대두되고 있다 [12,13]. 네트워크의 속도가 급속하게 발전하는 상황에서 많은 양의 데이터를 처리해야 하는 보안 시스템을 직접 사용해 보안 시스템의 성능을 평가하는 것은 많은 비용과 노력을 요구하므로 이를 효과적으로 해결하기 위한 대안이 시뮬레이션 모델을 통해 보안 시스템을 평가하는 것이다. 시뮬레이션 모델을 통해 구축된 시뮬레이션 환경은 다양한 환경을 조성하고, 시뮬레이션을 반복적으로 수행할 수 있으므로 변화하는 네트워크의 상황에 알맞은 보안 환경을 효과적으로 설정할 수 있다.

본 연구진은 침입 탐지 시스템에 전문가 시스템 (Expert System)을 적용하고 그 추론 과정에 있어서 효과적인 알고리즘을 선택하여 향상된 성능의 침입 탐지 시스템의 모델을 설계할 뿐만 아니라 네트워크에 분산된 침입 탐지 에이전트들과 침입 차단 시스템의 연동을 위하여 계약망 프로토콜을 적용한 시스템 환경을 구축할 것이다. 또한 이산 사건 시뮬레이션을 수행하기 위해 체계적으로 잘 정립된 이론인 DEVS 형식론으로 기존의 모델링 기법이 갖는 한계를 극복하여 대규모 시스템에 적합하도록 보안 시스템을 모델링하여 범용 보안 시뮬레이션 환경을 구축할 것이다 [14].

2. 보안 시뮬레이션 환경의 설계

2.1 네트워크 보안 모델의 설계

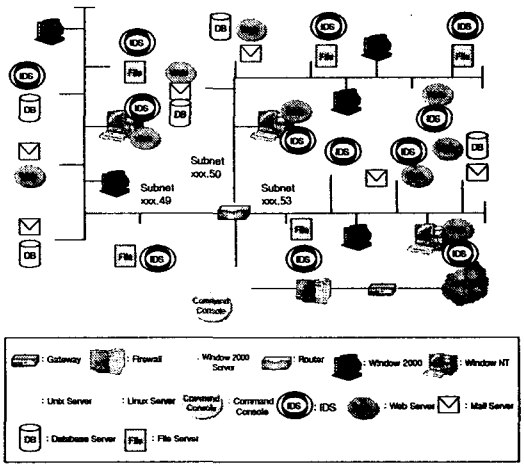
범용 보안 시뮬레이션 환경의 구축을 위해서 네트워크 및 보안 요소에 대한 모델링과 시뮬레이션을 수행할 네트워크의 설계가 선행되어야 한다.

2.1.1 대상 네트워크의 설계

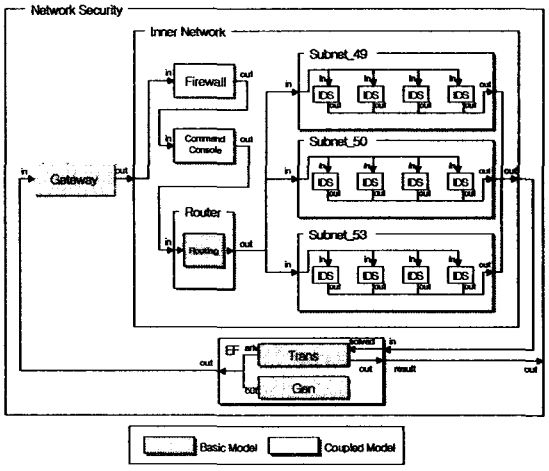
실시스템 수준의 시뮬레이션 환경 구축에 있어서 대상 네트워크의 설계는 시뮬레이션의 결과가 실시스템에 반영될 수 있는지를 판단하는 기준이 될 수 있다.

<그림 1>은 3개의 서브넷을 갖는 대상 네트워크의 구성도이다. 내부 네트워크에는 웹 서버, 메일 서버, 데이터베이스 서버 및 파일 서버를 설치한 호스트들이 있고 각 서브넷 별로 4개의 IDS가 장착되어 있다. 그리고 네트워크 구성요소로서 Router, Gateway, Firewall, Command Console이 구성되어 있다.

<그림 2>는 시뮬레이션을 수행하기 위해 DEVS 형식론에 근거하여 대상 네트워크를 모델링한 네트워크 보안 모델의 구조도이다.



<그림 1> 대상 네트워크 구성도

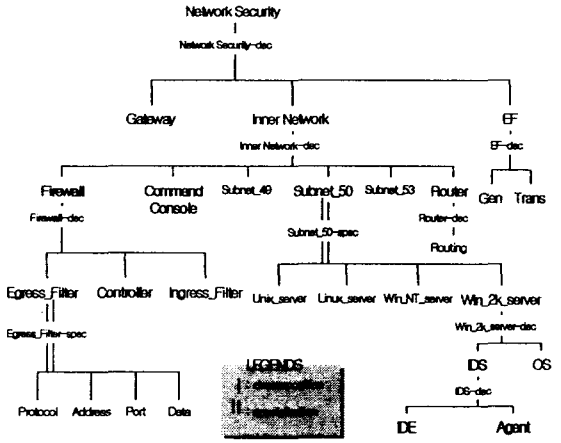


<그림 2> DEVS 시뮬레이션 모델의 구조도

Gateway 모델을 기준으로 EF 모델과 Inner Network 모델로 나눌 수 있는데 EF 모델은 Gen 모델과 Trans 모델로 구성된다. Gen 모델은 실제 네트워크에서 수집한 패킷 데이터를 바탕으로 시뮬레이션을 위한 패킷을 생성하여 Inner Network로 보낸다. Trans 모델은 시뮬레이션을 통해 얻은 결과를 분석하고 시뮬레이션을 통제한다.

2.2 대상 네트워크의 SES

<그림 3>은 대상 네트워크의 SES (System Entity Structure)를 나타낸 것이다. 각 모델들은 계층적으로 decomposition 및 specialization 관계를 가지고 있으며 Firewall, Command Console, IDS 모델은 다음 장에서 그 하위 모델과 기능에 대하여 자세하게 설명하도록 하겠다.



<그림 3> 대상 네트워크의 SES

3. 계약망 프로토콜의 설계

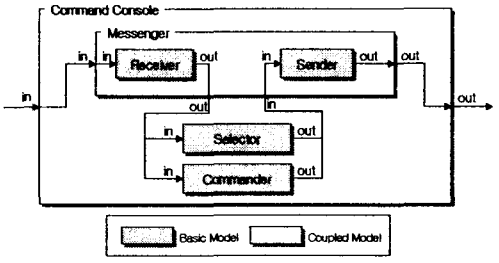
3.1 계약망 프로토콜

계약망 프로토콜은 분산된 문제를 해결하는데 있어 통신하고 조정하기 위한 도구로서 제안되었다 [10]. 계약망 프로토콜의 사용은 분산 감지 시스템과 분산 전달 시스템을 위해서 시도되었다 [15]. 계약망 프로토콜은 에이전트들이 계약 (contract)에 의하여 분산된 문제를 해결하기 위하여 협상하고 통신하는 메커니즘을 제공한다 [9]. 에이전트들은 수행될 필요가 있는 작업을 알리고 다른 에이전트들에 의해 공지된 작업들을 수행하기 위해 비드(bid)를 만들고, Command Console은 각 에이전트들이 제출한 비드를 평가하여 계약을 체결하게 된다 [9]. 계약망 프로토콜의 적용은 다중 침입 탐지 시스템에 있어서 서로 보완하고 협력하여 탐지의 성능을 향상시키고 정확도를 높일 수 있다.

3.2 Command Console 모델

계약망 프로토콜에서 모든 IDS 모델과 Firewall 모델의 에이전트들을 통제하게 되는 Command Console 모델의 모듈과 각 모듈의 기능은 다음과 같다.

Messenger 모델은 메시지의 송수신을 관리하는데 Receiver와 Sender로 구성된다. Receiver는 IDS에서 보낸 메시지를 받고 Sender는 Selector나 Commander에서 만들어진 메시지를 해당 IDS나 모든 IDS에게 unicast, multicast, broadcast의 방법으로 보낸다. Selector 모델은 모든 IDS가 보낸 비드로 내부 네트워크를 감시할 IDS를 선택한다. Commander 모델은 내부 네트워크의 상태에 따라 IDS와 Firewall를 통제하는 메시지를 결정한다.



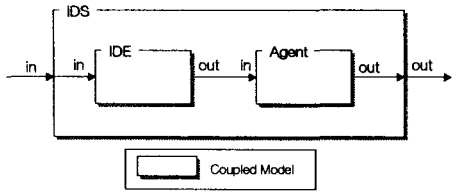
<그림 4> Command Console 모델의 구조도

3.3 IDS 모델

Command Console에서 오는 패킷 데이터는 IDE(Intrusion Detection Engine) 모델에서 탐지하게 되고 컨트롤과 관련된 메시지는 IDE 모델을 통과하여 Agent 모델에서 처리하게 된다. 또한 IDE 모델에서 탐지하거나 처리된 메시지를 Agent 모델로 보내 처리하게 된다.

<그림 5>는 IDE 모델과 Agent 모델의 커

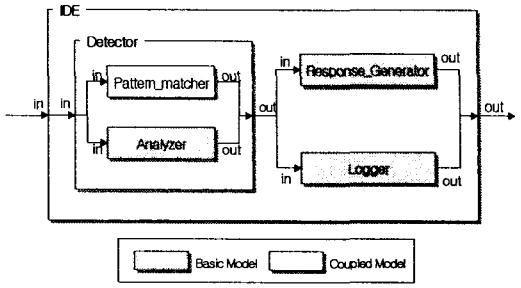
플링 구조를 나타내고 있다.



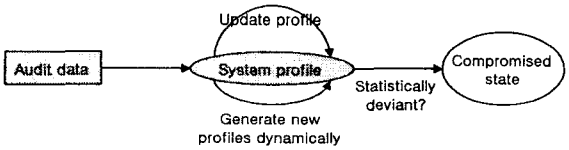
<그림 5> IDE와 Agent의 커플링 다이어그램

3.3.1 IDE 모델

IDE 모델은 <그림 6>과 같이 Detector 모델, Response_Generator 모델 그리고 Logger 모델로 구성되어 있다.



<그림 6> IDE 모델의 구조도

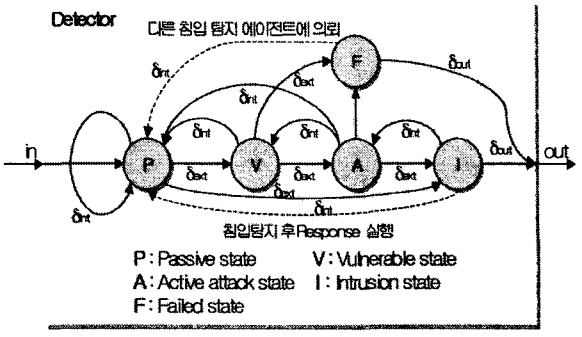


<그림 7> Analyzer 모델의 이상 탐지 과정

Detector 모델은 Pattern_matcher와 Analyzer로 구성되는데 Pattern_matcher는 규칙 베이스 전문가 시스템을 적용하여 입력된 패킷 데이터를 규칙과 패턴 매칭의 과정을 통해 침입을 탐지하게 된다. 무엇보다도 Rete 패턴 매칭 알고리즘[16]을 적용하여 전문가 시스템에서 가장 많은 시간을 소비하는 패턴 매칭의 시간을 단축시킬 수 있다. Analyzer는 통계적인 침입 탐지를 수행하는 모듈로 시스템 로그나 시스템 감사에 저장된 자료를 분석하여 침입을 탐지하게 된다. Response_Generator 모델은 Detector 모델에서의 침입 탐지 결과에 따라 IDS가 취할 행동을 결정하고 메시지를 보낸다. Logger 모델은 Detector 모델의 탐지 과정에서 발생하는 모든 정보를 로그로 기록한다.

Detector 모델은 상태전이를 통해 침입을 탐지하게 된다. <그림 8>은 Detector 모델의 상태전이도로 패킷 데이터가 입력되면 초기상태인 Passive 상태에서 시작되어 Vulnerable, Active attack 상태를 거쳐 목적상태인Intrusion 상태로 전이되어 침입을 탐지하게 된다. 침입을 탐지하는데 실패하거나 IDS가 설치된 시스템의 부하가 미리 정의된 임계값 이상인 경우에는 Failed 상태로 상태전이가 이루어지고 다른 IDS에게 침입 탐지를 의뢰하게 된다.

단일 사건 시그니처 (signature)의 경우는 Passive 상태에서 침입을 나타내는 하나의 사건 시그니처가 발생하게 되면 Intrusion 상태로 전이하여 침입을 탐지하게 되고 다중 사건 시그니처의 경우는 미리 정의한 사건 시그니처의 발생빈도나 진행 시나리오에 따라 Vulnerable, Active attack, Intrusion 상태로 전이하여 침입을 탐지하게 된다.

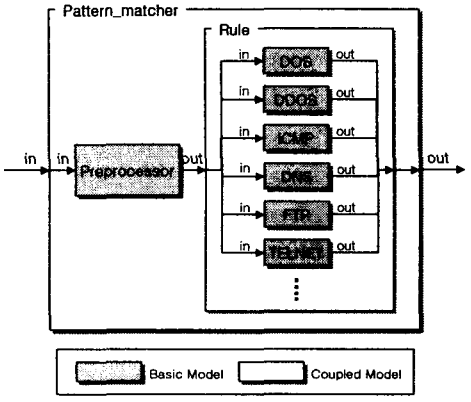


<그림 8> Detector 모델의 상태전이도

예를 들어 DDOS (Distributed Denial Of Service)의 일종인 Trinoo 공격을 탐지하는

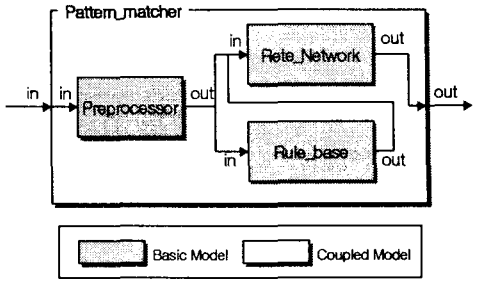
과정을 살펴보면 공격자가 마스터에게 default password인 gOrave를 보내면 Passive에서 Vulnerable 상태로 전이가 발생한다. 다시 default startup password인 betaalmostdone을 보내면 Vulnerable에서 Active attack 상태로 전이가 이루어진다. 마지막으로 default mdie password인 killme를 보내면 Intrusion 상태로 전이하여 침입을 탐지하고 Response _Generator 모델에게 알리고 Passive 상태로 돌아간다. DOS 공격인 Jolt 공격의 경우에는 공격 패킷이 유입되어 탐지 임계값을 넘게되면 Passive에서 Vulnerable, Intrusion 상태로 전이하여 침입을 탐지하게 된다.

<그림 9>는 Pattern_matcher 모델에 rete 알고리즘을 적용하지 않은 일반적인 구조를 나타낸다. Preprocessor 모델은 Rule 모델에서 사용하는 패킷 데이터를 미리 정의된 형태로 조작한다. Rule 모델은 유사한 시그리처를 가진 몇 개의 하위 모델로 이루어진다. 각 하위 모델은 패턴 매칭에 의해서 침입을 탐지하게 된다.



<그림 9> Patten_matcher 모델의 일반적 구조

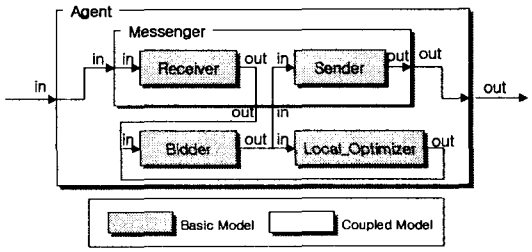
<그림 10>은 rete 알고리즘을 적용한 Pattern_matcher 모델의 구조도이다 Preprocessor 모델은 위와 같은 기능을 수행하고 Rete_Network 모델은 Rule_base 모델에서 만들어진 규칙으로 rete 네트워크를 생성한다.



<그림 10> rete 알고리즘을 적용한 Pattern_matcher 모델의 구조

3.3.2 Agent 모델

<그림 11>은 계약망 프로토콜에서 Command Console과 통신하게 되는 Agent 모델의 구조도이다. Agent 모델은 Messenger, Bidder 그리고 Local_Optimizer로 구성된다.



<그림 11> Agent 모델의 구조도

Messenger 모델은 Command Console에서 보낸 메시지를 받는 Receiver와 Bidder에 의해 만들어진 메시지를 보내는 Sender로 구성되어 메시지의 송수신을 관리한다. Bidder 모델은 Local_Optimizer 모델과 통신하여 bid를 만들고, IDE 모델을 Command Console의 명령에 따라 통제한다. Local_Optimizer 모델은 시스템 상태나 IDE에서 적용되는 규칙과 같이 bid를 위해 제공되는 정보를 관리하고 최적화한다. expertise, experience, loading에 대한 상태정보를 관리하는데 expertise는 해당 IDS의 규칙의 개수를 나타내고 experience는 해당 IDS가 이전에 침입을 탐지한 횟수이다. 만약 침입 탐지 Agent가 침입을 탐지한 경우에는

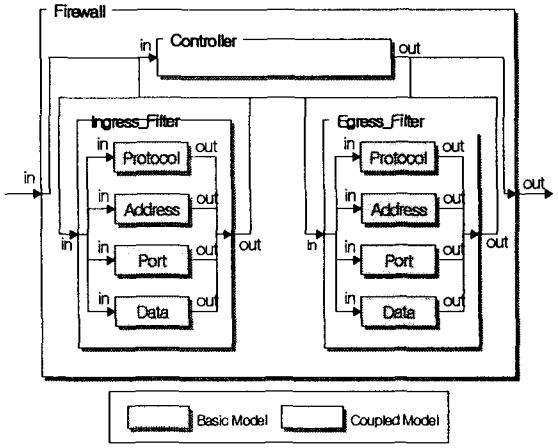
experience 값을 1만큼 증가시킨다. loading의 경우에는 시스템 부하에 따라 three-level scheme를 적용하는데 under-loaded, normal, over-loaded의 3단계로 구분하여 Bidder에게 정보를 제공하게 된다.

3.4 Firewall 모델

Firewall 모델은 기본적으로 내부 네트워크로 유입되거나 내부 네트워크에서 나가는 패킷에 대한 필터링을 수행한다. 또한 IDS에서 침입을 탐지했을 경우에 Command Console과 통신하여 침입에 해당하는 패킷을 차단하거나 Response에 해당하는 동작을 취하게 된다.

대상 네트워크의 구조상 Command Console이 모든 Agent를 통제하기 때문에 Command Console이 마비되면 모든 시스템의 기능이 정지된다. 따라서 Firewall 모델에서는 외부에서 Command Console에 접근하는 모든 패킷을 필터링하고 내부적으로 Agent 모델에서 오는 메시지를 제외한 모든 패킷을 필터링해야 한다.

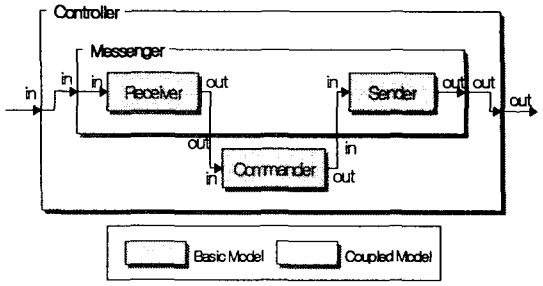
<그림 12>에서 Firewall 모델은 Controller, Ingress_Filter 그리고 Egress_Filter로 구성되어 있다. Controller 모델은 Firewall 모델의 내부에 있는 Ingress_Filter와 Egress_Filter를 통제하고 Command Console 모델에서 오는 모든 메시지를 관리한다. Ingress_Filter 모델은 외부에서 내부 네트워크로 유입되는 패킷을 필터링하고 Protocol, Address, Port 그리고 Data로 구성된다. Protocol은 패킷에서 추출된 프로토콜에 대한 정보를 필터링하고 Address는 IP 주소에 대한 필터링을 수행하며 Port는 패킷에서 포트에 대한 필터링을 수행한다. Data는 패킷의 데이터 부분에 대한 필터링을 한다. Egress_Filter 모델은 내부에서 외부로 나가는 패킷을 필터링하고 Ingress_Filter와 동일한 하위 모델을 갖는다.



<그림 12> Firewall 모델의 구조도

3.4.1 Controller 모델

<그림 13>은 Firewall 모델의 내부에 있는 Controller 모델의 구조도이다. Controller 모델은 Messenger와 Commander로 구성되어 있고 Messenger 모델은 외부나 내부 네트워크에서 유입되는 패킷, Command Console에서 오는 메시지 그리고 Ingress_Filter 및 Egress_Filter에서 오는 모든 메시지를 관리한다. Commander 모델은 실질적으로 Firewall 모델이 취하는 모든 동작을 판단하고 통제하며 Command Console에서 오는 메시지를 통해 Firewall 모델의 동작을 통제한다.

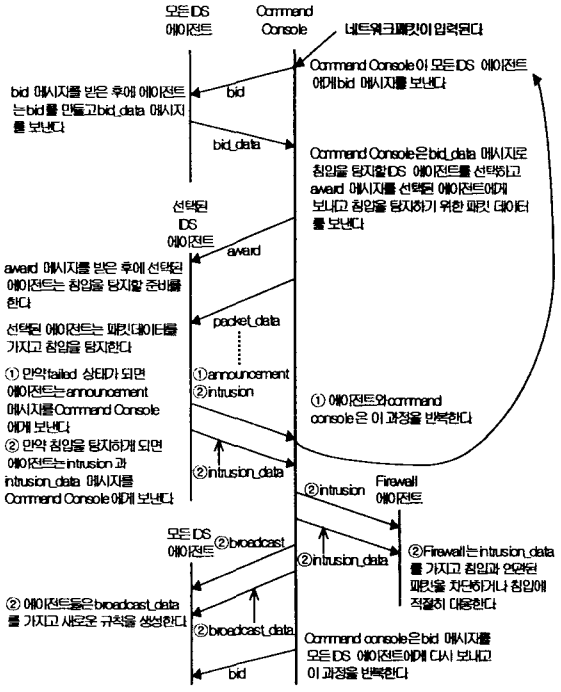


<그림 13> Controller 모델의 구조도

4. 계약망 프로토콜을 통한 연동

4.1 계약망 프로토콜의 연동과정

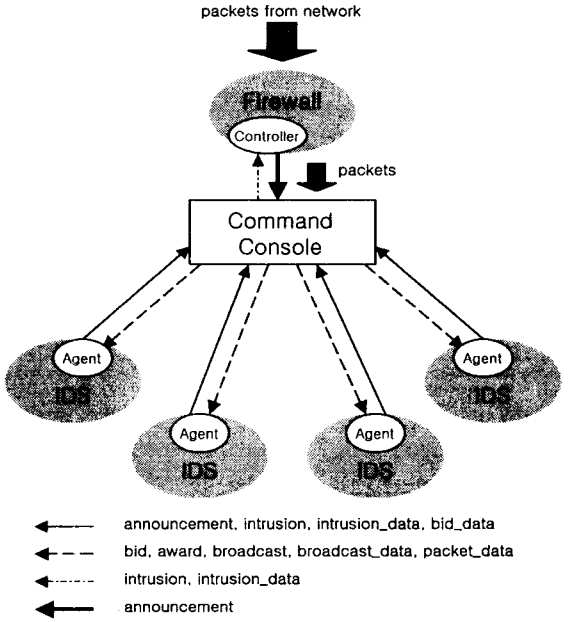
시뮬레이션이 시작되고 내부 네트워크로 패킷이 유입되면 Command Console은 모든 침입 탐지 Agent에게 bid 메시지를 보내고 이 메시지를 받은 Agent들은 bid_data 메시지를 보낸다. Command Console은 bid_data를 가지고 선택 알고리즘에 의해 침입을 탐지할 Agent를 선택하게 되고 선택된 Agent에게 award 메시지를 보낸다. award 메시지를 받은 Agent는 packet_data를 기다리고 Command Console은 패킷 정보를 packet_data에 복사하여 선택된 Agent에게 보낸다. 선택된 Agent는 이 데이터를 가지고 침입을 탐지하게 된다.



<그림 14> 메시지를 교환하는 연동과정

침입 탐지 과정 중 Detector 모델에서 상태전이가 발생하여 failed 상태가 되면 Agent는 Command Console에 announcement 메시지를 보내고 이 메시지를 받은 Command Console은 위의 과정과 마찬가지로 다른 모든 Agent에게 bid 메시지를 보내고 Agent 선택과정을 반복하게 된다.

침입을 탐지한 경우에는 선택된 Agent가 intrusion 메시지를 Command Console에 보내고 intrusion_data 메시지를 보낸다. 이 메시지를 받은 Command Console은 Firewall에게 intrusion과 intrusion_data 메시지를 차례로 보내고 모든 침입 탐지 Agent에게 broadcast 메시지와 침입에 대한 정보를 broadcast_data 메시지로 보낸다. 그런 다음 다시 bid 메시지를 보내고 위의 Agent 선택과정을 반복한다. <그림 14>는 IDS, Command Console, Firewall 모델들이 메시지를 교환하는 연동과정을 순차적으로 표현하였다.



<그림 15> CNP의 연동구조와 메시지의 흐름

<그림 15>는 계약망 프로토콜의 연동구조와 메시지의 흐름을 나타낸다. Command Console은 계약망 프로토콜에서 IDS와 Firewall를 중앙에서 통제하는 중요한 역할을 수행하게 된다. 실제적으로 연동은 Command

Console과 IDS 내부의 Agent 모델, Firewall 내부의 Controller 모델사이의 메시지 교환에 의해 이루어진다.

4.2 메시지(Message)

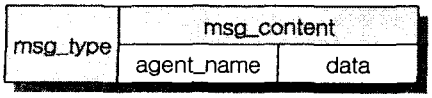
Command Console과 IDS 및 Firewall 모델들은 모두 Messenger 모델을 가지고 있는데 이 모델은 메시지의 송신과 수신을 담당한다. 메시지의 종류는 크게 컨트롤 메시지 (control message)와 데이터 메시지 (data message)로 구분되며 각각의 메시지는 msg_type 필드의 값에 의해 구분된다. 메시지의 구조와 종류는 <표 1>과 같다.

<표 1> 메시지의 구조와 종류

msg_type	msg_content

	msg_type	메시지 종류
Control Message	0	Broadcast
	1	Announcement
	2	Bid
	3	Award
	4	Intrusion
Data Message	5	broadcast_data
	6	bid_data
	7	packet_data
	8	intrusion_data

<그림 16>은 msg_content의 구조를 나타낸다. 메시지를 보내거나 받을 때 각 에이전트들을 구별하고 그 주소를 판별할 수 있는 agent_name 필드와 메시지의 종류에 따라 갖는 data 필드로 구성된다. Command Console에는 agent_name과 에이전트의 위치를 나타내는 매핑 테이블이 있어서 메시지를 받거나 보내게 된다.



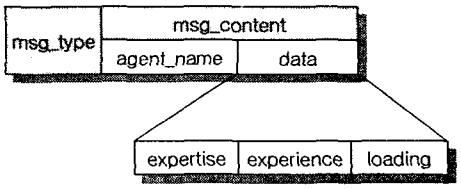
<그림 16>msg_content의 구조

4.2.1 컨트롤 메시지(Control Message)

컨트롤 메시지에는 다음의 다섯 가지가 있다. bid 메시지는 비드를 제출하도록 모든 침입 탐지 에이전트들에게 알린다. award 메시지는 Command Console에서 선택된 에이전트에게 침입 탐지 에이전트로 선택된 것을 알린다. intrusion 메시지는 침입이 발생하거나 상태전이로 intrusion 상태가 되면 침입 탐지 에이전트가 Command Console에 알린다. 또한 침입이 탐지되었을 때 침입 차단 시스템에 알려서 침입에 대응하게 한다. announcement 메시지는 상태전이로 failed state가 되어 에이전트의 선택을 다시 해야할 경우 Command Console에 알린다. broadcast 메시지는 침입 탐지 에이전트가 침입을 탐지하면 탐지에 대한 정보를 받은 Command Console이 모든 침입 탐지 에이전트에게 탐지 정보를 보낸다는 것을 알린다.

4.2.2 데이터 메시지(Data Message)

데이터 메시지는 네 가지가 있다. bid_data 메시지는 bidding 과정에서 에이전트를 선택하기 위해 필요한 정보를 가지고 있고 intrusion_data 메시지는 탐지된 침입의 패킷 정보를 가지고 있다. packet_data는 네트워크로 유입되는 패킷 데이터를 가지고 있으며 broadcast_data는 탐지된 침입에 관련된 정보를 포함한다.

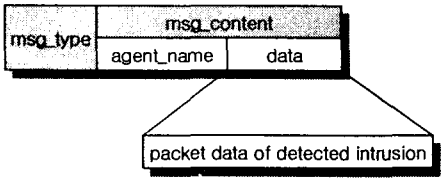


<그림 17> bid_data의 구조

데이터 메시지 중에서 bidding 과정을 통해 Agent를 선택하는 bid_data 메시지는 <그림 17>과 같이 data 필드가 expertise, experience, loading으로 구성된다. expertise는 각 Agent

가 탐지할 수 있는 규칙과 전문성을 수치화한 것이며 experience는 이전에 탐지한 경험이 있는 침입에 대한 것을 수치화한 것이다. 그리고 loading은 침입 탐지 시스템을 장착한 시스템의 CPU와 주 메모리에 대한 부하를 수치화한 것이다. 이 세 가지 수치는 Local _Optimizer 모델에서 관리한다.

또 다른 데이터 메시지 중에서 침입 탐지 시스템이 탐지한 내용을 다른 Agent들에게 알리는 broadcast_data 메시지는 <그림 18>과 같이 data 필드에는 탐지된 침입에 대한 패킷 정보를 포함하게 된다.



<그림 18> broadcast_data의 구조

4.3 침입 탐지 에이전트 선택 알고리즘

각 에이전트는 loading 필드의 값이 임계값을 넘지 않은 경우에 bid_data를 Command Console에 보내게 되는데 첫 번째로 expertise 필드의 값을 기준으로 정렬하여 가장 큰 값을 갖는 에이전트를 선택한다. 만약 같은 값을 갖는 에이전트가 존재하면 그 에이전트 중에 experience 필드의 값을 기준으로 다시 정렬하여 역시 가장 큰 값을 가진 에이전트를 선택한다. 만약 experience 값마저 같은 에이전트가 존재한다면 마지막으로 loading 필드의 값을 기준으로 정렬하여 가장 작은 값을 가진 에이전트를 선택하게 된다. 다음은 에이전트 선택 알고리즘이다.

```
Let bidi be bids
Set bid_list = empty set
Let bid_list = ( bid1, bid2,..., bidn ) be a list of bids
for i = 1 to n
  if loading of bidi >= threshold value
    Delete bidi from bid_list
Sort bid_list by expertise in descending order
if the number of bid including the greatest value of expertise>2 then
{
  Delete bids from bid_list except bids including the greatestvalue of expertise
  Sort bid_list including bids of the same expertise by experince in descending order
  if the number of bid including the greatest value of experience >= 2 then
  {
    Delete bids from bid_list except bids including the greast value of experience
    Sort bid_list including bids of the same experience by loading in ascending order
  }
}
Select Agent from bid_list(the first element)
```

<그림 19> 에이전트 선택 알고리즘

5. Rete 패턴 매칭 알고리즘의 적용

5.1 침입 탐지 전문가 시스템

침입을 탐지하는 전문가 시스템은 규칙의 집합인 지식 베이스, 추론을 수행하는 추론 엔진 그리고 사실을 저장하는 working memory (WM)로 구성된다. 추론 방식은 전향 추론 방식 (forward chaining inferencing)을 적용하는데 추론과정은 패턴 매칭 (pattern matching), 충돌 해결 (conflict resolution) 그리고 실행 (act)의 순서로 이루어진다 [16]. 각 규칙의 left-hand side (LHS)는 규칙에서 if 부분과 일치하는 조건의 결합으로 구성되고 right -hand side (RHS)는 규칙에서 then 부분과 일치하는 일련의 행동들로 구성된다. 패턴 매칭은 규칙에서 LHS와 WM에 있는 사실들을 비교한다. 이 과정의 결과 만족된 규칙이 2개 이상인 경우는 충돌 셋 (conflict set)이 구성된다. 다음에는 미리 정의된 충돌 해결 전략 (conflict resolution policy)에 따라 충돌 셋에서 하나의 규칙을 선택하고 마지막으로 그 규칙의 일련의 행동들를 실행하여 WM에 있는 내용을 변화시킨다. WM는 초기상태에서 추론과정을 통해 규칙을 선택하고 최종적으로 침입을 탐지하게 된다.

다음은 침입 탐지 시스템이 추론과정에서 사용할 몇 가지 침입 탐지 규칙에 대한 예를 든 것이다. 규칙들은 DOS, DDOS 등과 같이 각 규칙의 모듈별로 탐지가 이루어진다.

```
DOS Rules :
R1 :
if protocol == ip ∧ IPSrcAddr == EXTERNAL_NET ∧ IPDstAddr == HOME_NET ∧
fragbits == "M" ∧ dsize == 408 ∧ state == "passive"
then state = "vulnerable", p_count+=1
R2 :
if protocol == ip ∧ IPSrcAddr == EXTERNAL_NET ∧ IPDstAddr == HOME_NET ∧
fragbits == "M" ∧ dsize == 408 ∧ state == "vulnerable"
then p_count+=1
R3 :
if protocol == ip ∧ IPSrcAddr == EXTERNAL_NET ∧ IPDstAddr == HOME_NET ∧
fragbits == "M" ∧ dsize == 408 ∧ state == "vulnerable" ∧ p_count >= threshold
then state = "intrusion", msg("DOS Jolt attack")
```

<그림 20> DOS 규칙의 예

```
ICMP Rules :
R15 :
if protocol == icmp ∧ IPSrcAddr == EXTERNAL_NET ∧ IPDstAddr == HOME_NET
∧ data == "|5768 6174 7355 7020 2d20 4210 4e65 7477|" ∧ itype == 8
then state = "intrusion", msg("ICMP PING WhatsupGold Windows)
R16 :
if protocol == icmp ∧ IPSrcAddr == EXTERNAL_NET ∧ IPDstAddr == HOME_NET
∧ dsize > 800
then state = "intrusion", msg("ICMP Large ICMP Packet")
```

<그림 21> ICMP 규칙의 예

```
DDOS Rules :
R11 :
if protocol == udp ∧ IPSrcAddr == EXTERNAL_NET ∧ IPDstAddr == HOME_NET
∧ DstPort == 31335 ∧ data == "*HELLO*" ∧ state == "passive"
then state = "vulnerable", msg("DDOS
Trin00/:DaemontoMaster(*HELLO*detected)")
R12 :
if protocol == udp ∧ IPSrcAddr == EXTERNAL_NET ∧ IPDstAddr == HOME_NET
∧ DstPort == 31335 ∧ data == "PONG" ∧ state == "vulnerable"
then state = "intrusion", msg("DDOS Trin00/:DaemontoMaster(PONGdetected)")
R13 :
if protocol == tcp ∧ IPSrcAddr == EXTERNAL_NET ∧ IPDstAddr == HOME_NET
∧ DstPort == 27665 ∧ TCPACKFlag == true ∧ data == "gOrave" ∧ state ==
"passive"
then state = "vulnerable", msg("DDOS Trin00/:Attacker to Master default
password")
R14 :
if protocol == tcp ∧ IPSrcAddr == EXTERNAL_NET ∧ IPDstAddr == HOME_NET
∧ DstPort == 27665 ∧ TCPACKFlag == true ∧ data == "betaalmostdone" ∧
state == "vulnerable"
then state = "active attack", msg("DDOS Trin00/:Attacker to Master default
startup password")
R15 :
if protocol == tcp ∧ IPSrcAddr == EXTERNAL_NET ∧ IPDstAddr == HOME_NET
∧ DstPort == 27665 ∧ TCPACKFlag == true ∧ data == "killme" ∧ state ==
"active attack"
then state = "intrusion", msg("DDOS Trin00/:Attacker to Master default mdie
password")
```

<그림 22> DDOS 규칙의 예

```
Goal Rule :
if state = "intrusion" then passivate(), p_count = 0
```

<그림 23> Goal 규칙의 예

각 규칙들은 조건을 나타내는 if 부분과 행동을 나타내는 then 부분으로 구성되어 있고 침입을 탐지한 이후에 Goal Rule을 실행하여 다시 초기 상태로 전이하게 된다. 위의 규칙에서 threshold는 침입에 대한 상태전이를 발생시키는 탐지 임계값을 의미하고 탐지 임계값을 변화시키며 시뮬레이션을 수행하면 효과적인 침입 탐지를 위한 탐지 임계값의 설정이 가능하다.

5.2 Rete 패턴 매칭 알고리즘

Rete 알고리즘은 규칙이 선택된 후에 충돌셋을 다시 계산하기 위해서 요구되는 노력을 감소시키는 것에 의해 전향 추론 시스템(forward-chained rule system)의 속도를 향상시키는 알고리즘이다 [16]. Rete 알고리즘을 활용하면 침입 탐지 전문가 시스템의 추론과정에서 가장 많은 시간을 소비하는 패턴 매칭의 시간을 감소시켜 침입 탐지의 성능을 향상시킬 수 있다.

5.3 Rete 네트워크

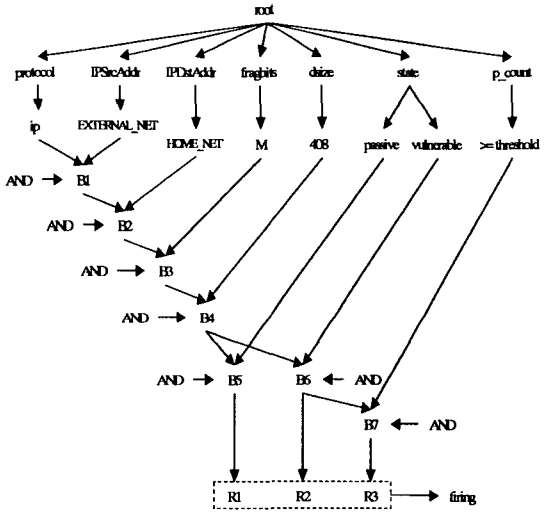
Rete는 거대한 집합의 패턴을 거대한 집합의 오브젝트에 반복 없이 비교하는 매칭 알고리즘이다 [17]. Rete 알고리즘은 노드 (node)로 구성된 네트워크를 만드는 것에 의해서 구현되는데 root 노드를 제외한 노드들은 패턴을 나타내고 root에서 leaf까지의 경로는 규칙의 LHS를 나타낸다. 지식 베이스로부터 첨가되거나 제거되는 사실들은 노드로 구성된 네트워크에 의해 진행된다. 네트워크에서 가장 아래에 있는 터미널 노드들은 개별적인 규칙을 나타내는데 사실의 집합이 Rete 네트워크의 터미널 노드에 이르렀을 때 특정한 규칙의 LHS에 있는 모든 테스트를 통과한 것이며 선

택된 규칙의 RHS를 실행하게 된다.

Rete 네트워크에는 크게 하나의 입력이 있는 노드 (one-input alpha node; Alpha memory)와 두 개의 입력이 있는 노드 (two-input beta node; Beta memory)가 있다. 하나의 입력이 있는 노드는 개별적인 사실에 대한 테스트를 수행하여 알파 메모리에 저장하고 두 개의 입력이 있는 노드는 사실들 상호간의 테스트를 수행하는 동시에 그룹핑 함수를 수행하여 베타 메모리에 저장한다.

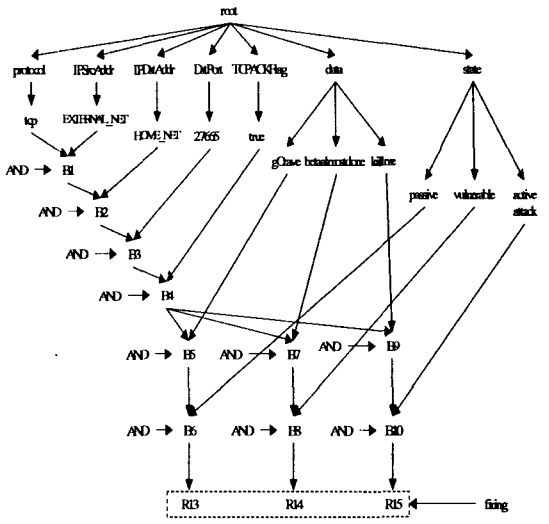
이 시점에서 침입 탐지 전문가 시스템에 적용된 몇 가지 규칙들을 Rete 알고리즘을 이용하여 구성한 Rete 네트워크는 다음과 같다.

<그림 24>는 DOS Jolt 공격을 탐지하는 규칙을 Rete 네트워크로 구성하였다. root 노드로부터 터미널 노드인 규칙까지 알파 노드와 베타 노드를 AND 연산에 의해 그룹핑하여 구성하였으며 패킷 정보가 root 노드에 입력되면 각 패턴에 따라 노드를 이동하고 터미널 노드에 이르게 되면 해당 규칙을 적용하게 된다.



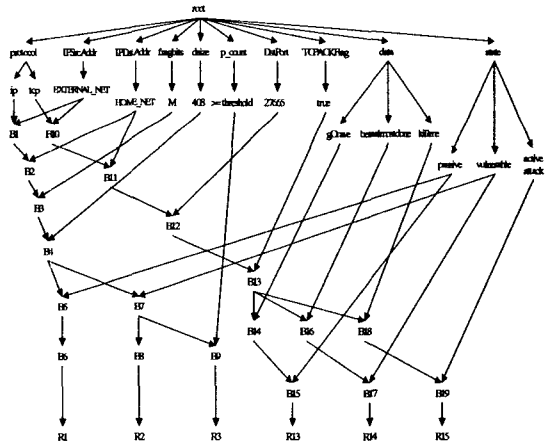
<그림 24> Rete 네트워크의 예제1

<그림 25>는 DDOS Trinoo 공격을 탐지하는 규칙을 Rete 네트워크로 구성하였다.



<그림 25> Rete 네트워크의 예제2

<그림 26>은 예제1과 예제2의 규칙을 통합하여 하나의 Rete 네트워크를 구성하였다. 이와 같은 방법으로 DOS, DDOS 등과 같이 모듈화된 침입 탐지 전문가 시스템에 Rete 알고리즘을 적용하여 하나의 Rete 네트워크를 구성하고, Rete 네트워크에서 root로부터 터미널 노드까지의 경로를 따라 추론을 수행하여 침입을 탐지하게 된다.



<그림 26> Rete 네트워크 구성(예제1+예제2)

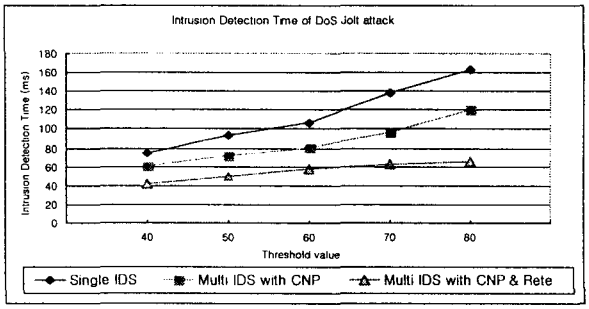
6. 시뮬레이션

6.1 성능 지표의 설정

시뮬레이션의 결과를 평가하기 위하여 침입 탐지 시간과 침입 오판율을 성능 지표로 설정하였다. 침입 탐지 시간은 패킷 데이터 상에서 발생하는 침입을 침입 탐지 시스템이 탐지하는데 경과하는 시간을 나타낸다. 침입 오판율은 false-positive와 false-negative로 구분하는데 false-positive 오판율은 침입이 아닌 것을 침입으로 판단하는 경우의 정도를 나타내고 false-negative 오판율은 침입인 것을 침입으로 탐지하지 못하는 경우의 정도를 나타낸다.

6.2 시뮬레이션 결과 및 분석

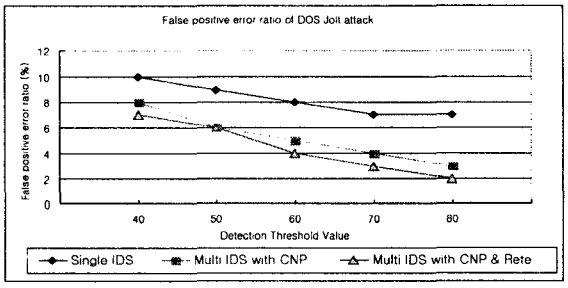
본 연구진은 단일 침입 탐지 시스템과 계약망 프로토콜을 적용한 다중 침입 탐지 시스템 및 Rete 알고리즘과 계약망 프로토콜을 적용한 다중 침입 탐지 시스템에 대한 시뮬레이션을 수행하였다.



<그림 27> DOS Jolt 공격의 침입 탐지 시간

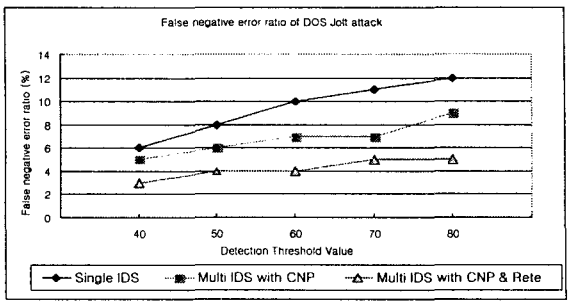
<그림 27>은 DOS Jolt 공격의 탐지를 위한 탐지 임계값이 40, 50, 60, 70, 80으로 증가함에 따라 세 가지 경우의 침입 탐지 시간의 변화를 나타낸다. 여기서 DOS Jolt 공격은 표준에 규정된 길이 이상으로 큰 IP 패킷을 전송함으로써 이 패킷을 수신하는 운영체제에서 이 비정상적인 패킷을 처리하지 못함으로써 서비스거부공격을 유발하도록 하는 방법이다. 시뮬레이션 수행 결과, 계약망 프로토콜을 적용한 다중 침입 탐지 시스템이 단일 침입 탐지 시스템보다 더 빠르게 침입을 탐지했으며 Rete 알고리즘을 적용하면 탐지 속도가 더 향상되었다.

<그림 28>은 침입 탐지의 성능 지표인 false positive 오판율을 탐지 임계값이 증가함에 따라 나타낸 것으로 단일보다 다중 침입 탐지 시스템의 성능이 뛰어나고 Rete 알고리즘을 적용하면 false positive 오판율을 감소시킬 수 있다. 또한 임계값의 증가에 따라 침입이 아닌 것으로 침입으로 판단하는 비율이 낮아진다.



<그림 28> DOS Jolt 공격의 false positive 오판율

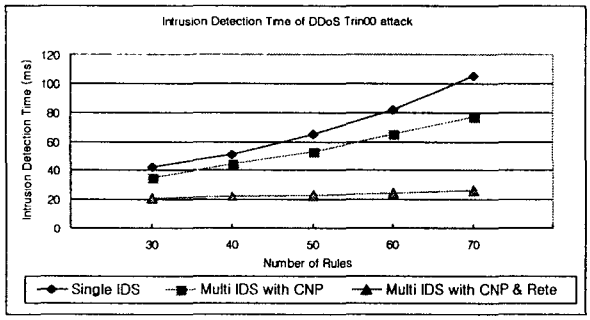
<그림 29>는 false negative 오판율을 탐지 임계값이 증가함에 따라 측정한 것으로 단일보다 다중 침입 탐지 시스템의 오판율이 낮았으며 Rete 알고리즘을 적용하면 성능이 향상되는 것을 알 수 있다. 그리고 임계값의 증가에 따라 침입을 침입으로 탐지하지 못하는 비율이 높아진다.



<그림 29> DOS Jolt 공격의 false negative 오판율

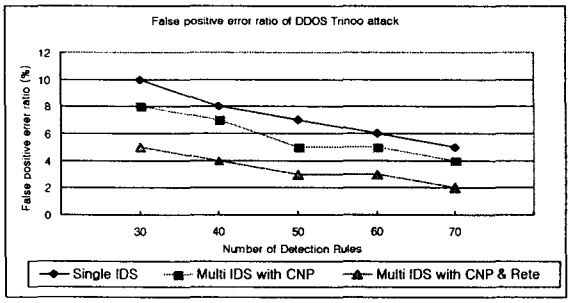
Trinoo 공격은 많은 소스로부터 통합된 UDP flood 서비스거부공격을 유발하는데 사용되는 도구로서 몇 개의 마스터들과 많은 수의 데몬들로 이루어진다.

Pattern_matcher 모델에서 DOS, DDOS, ICMP와 같은 탐지 모듈을 구성하는 규칙의 수는 침입 탐지 시스템이 침입을 탐지하는데 소요되는 패턴 매칭의 시간 및 오판율과 연관되어 있으며 같은 규칙을 가지고 Rete 알고리즘을 적용하여 Pattern_matcher를 구성한 경우에도 마찬가지 연관성을 갖는다.



<그림 30> DDOS Trinoo 공격의 침입 탐지 시간

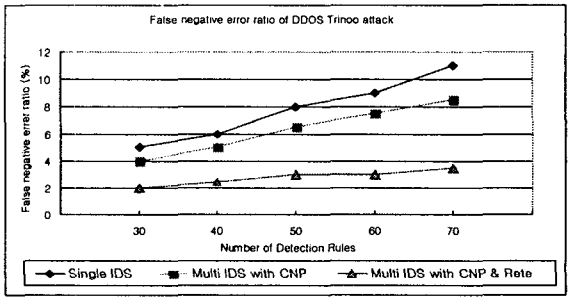
<그림 30>은 DDOS Trinoo 공격의 침입 탐지 시간을 규칙의 수가 증가함에 따라 나타낸 것이다. 단일보다 다중 침입 탐지 시스템의 탐지 시간이 더 빠르며 Rete 알고리즘을 적용하면 성능이 크게 향상되며 규칙의 수에 관계없이 거의 일정한 탐지 시간을 나타낸다.



<그림 31> DDOS Trinoo 공격의 false positive 오판율

<그림 31>은 DDOS Trinoo 공격의 false positive 오판율을 규칙의 수가 증가함에 따라 나타낸 것이다. Rete 알고리즘을 적용한 다중 침입 탐지 시스템의 경우 최소의 오판율을 가지며 규칙의 수에 크게 영향을 받지 않는 것으로 나타났다.

<그림 32>는 false negative 오판율을 규칙의 수가 증가함에 따라 나타낸 것이다. 단일보다는 계약망 프로토콜을 적용한 다중 침입 탐지 시스템의 성능이 우수하며 Rete 알고리즘을 적용한 경우 그 성능이 탁월하게 향상되는 것을 알 수 있다.



<그림 32> DDOS Trinoo 공격의 false negative 오판율

7. 결론

앞으로 네트워크의 활용은 더욱 증가할 것이며 네트워크에서 교환되는 정보의 가치와 중요성 또한 증가할 것이다. 반면 네트워크를 악용하는 정보 유출이나 침입 사고의 발생 역시 증가할 것이다. 이러한 상황에서 침입 탐지 시스템은 하나의 보안 요소가 될 것이며 그 효용성과 필요성이 평가될 것이다. 본 연구진은 시뮬레이션을 통하여 몇 가지 성능 지표를 설정하고 네트워크 보안을 위한 DEVS 기반의 네트워크 보안 시뮬레이션 환경을 구축하였으며 다중 침입 탐지 시스템과 침입 차단 시스템의 연동을 위해 계약망 프로토콜을 적용하였다. 또한 실질적으로 침입을 탐지하게 되는 침입 탐지 전문가 시스템에 Rete 패턴

매칭 알고리즘을 적용하여 성능을 평가하였다. 시뮬레이션을 통하여 단일 침입 탐지 시스템보다는 여러 개의 침입 탐지 시스템이 계약망 프로토콜에 의해 연동하는 다중 침입 탐지 시스템이 효과적으로 침입을 탐지하였으며 더 나아가 침입 차단 시스템과의 연동을 통해 네트워크를 강력하게 보호할 수 있다. 또한 침입 탐지 전문가 시스템의 추론 과정에 Rete 패턴 매칭 알고리즘을 적용하면 계약망 프로토콜을 적용한 다중 침입 탐지 시스템과 침입 차단 시스템의 성능을 월등히 향상시킬 수 있다.

향후 과제로는 다양한 보안 시뮬레이션을 수행할 수 있는 범용 네트워크 보안 시뮬레이션 환경의 구축이 필요하며 침입을 탐지하는 추론 과정에 적용될 효과적인 알고리즘의 계발이 필요할 것이다.

참고문헌

[1] Shan Zheng, Chen Peng, Xu Ying, Xu Ke, "A network state based intrusion detection model", Computer Networks and Mobile Computing 2001 Proceedings. 2001 International Conference, pp. 481-486, 16-19 Oct. 2001.

[2] R. Base. "Intrusion Detection", Macmillan Technical Publishing, 2000.

[3] E. Amoroso, "Intrusion Detection - An Introduction to Internet Surveillance, Correlation, Traps, Trace Back, and Response", Intrusion.Net Books, 1999.

[4] Duan Haixin, Wu Jianping, Li Xing, "Policy based access control framework for large networks", Proceedings. IEEE International Conference on ICON 2000, Sept. 2000.

[5] Noureldien A. Noureldien, Izzeldin M. Osman, "On Firewalls Evaluation Criteria", Proceeding of TENCON 2000, pp. 104-110, Sept. 2000.

[6] K. M. Sim, S. K. Shiu, and B. L. Martin, "Simulation of a Multi-agent Protocol for Task Allocation in Cooperative Design," IEEE SMC '99 Conference Proceedings. International Conference on, vol.3, pp. 95-100, 1999.

[7] Shungeng Hu, Li Zhang and Yixin Zhong, "Theories, Technology and Application of Multi-Agent Systems," Computer Science, Vol. 26, No.9, pp. 20-24, 1999.

[8] T. Sandholm, "An Implementation of the Contract Net Protocol based on Marginal Cost Calculations," in 11th National Conference on Artificial Intelligence (AAAI-93), Washington. DC, 1993.

[9] Jihoon Yang, Raghu Havaldar, Vasant Honavar, Les Miller and Johny Wong, "Coordination of Distrbuted Knowledge Networks Using Contract Net Protocol," Information Technology Conference, IEEE, pp. 71-74, 1998.

[10] R. Smith, "The Contract Net Protocol: High-level Communication and Control in a distributed problem solver," IEEE Transactions on Computers, vol. C-29, no. 12, pp. 1104-1113, December. 1980.

[11] H. van Dyke Parunak. Manufacturing Experince with the Contract Net. In Research Notes in Artificial Intelligence: Distributed Artificial Intelligence, Vol. 1, pp. 285 - 310, Morgan Kaufmann Publishers, 1987.

[12] Tae H. Cho, Bernard P.Zeigler, "Simulation of Intelligent Hierarchical Flexible Manufacturing : Batch Job Routing in Operation Overlapping", IEEE Transactions on Systems, Man and Cybernetics-PART A : System and Humans, vol. 27, no.1, p.116-126, Jan.,

1997.

[13] Zegler, B.P, Cho, T.H. and Rozenblit, J.W., "Knowledge Based System for Hierarchical Flexible Manufacturing System Modeling", IEEE Transactions on Systems, Man and Cybernetics -PART A, vol 26, no.1, p.81-89, Jan., 1996.

[14] H.S. Seo and T.H. Cho, "An application of blackboard architecture for the coordination among the security systems", Simulation Modelling Practice and Theory, Elsevier Science B.V., vol. 11, issues 3-4, pp. 269-284, Jul. 2003.

[15] T. Sandholm "An implementation of the contract net protocol based on marginal cost calculations", in 11th National Conference on Artificial Intelligence (AAAI-96), Washington. DC, 1993.

[16] V. Devedzic, D. Velasevic, "An architecture for real-time inference engines on personal computers", System Sciences, 1992. Proceedings of the Twenty-Fifth Hawaii International Conference on, vol. 1, pp. 619-630, 7-10 Jan. 1992.

[17] C. L. Forgy, "Rete: A fast algorithm for the many pattern/many object pattern match problem", Artificial Intelligence, vol. 19, pp. 17-37, 1982.

주 작 성 자 : 서 경 진
논문투고일 : 2003. 09. 18
논문심사일 : 2003. 11. 11(1차), 2003. 11. 11(2차), 2003. 12. 16(3차)
심사판정일 : 2003. 12. 16

● 저자소개 ●

서경진

2001 성균관대학교 전기전자 및 컴퓨터공학부 학사
2003 성균관대학교 정보통신공학부 컴퓨터공학과 석사과정
관심분야: 침입 탐지 시스템, 네트워크 보안, 보안시뮬레이션, 분산 시스템

조대호

1983 성균관대학교 전자공학과 학사
1987 University of Alabama 전자공학과 석사
1993 University of Arizona 전자 및 컴퓨터공학 박사
1993 ~1995 경남대학교 전자계산학과 전임강사
1995 ~1999 성균관대학교 전기전자 및 컴퓨터공학부 조교수
1999 ~2002 성균관대학교 전기전자 및 컴퓨터공학부 부교수
2002 ~현재 성균관대학교 정보통신공학부 부교수
관심분야 : 모델링 및 시뮬레이션, 네트워크 보안, 지능 제어, ERP