情報保護學會論文誌
第 11 券 第 4 號, 2001. 8

보안스킴의 자동확장성을 지원하는 미행 메커니즘*

장 희 진**, 김 상 욱**

A Shadowing Mechanism supporting Automatic Extension of Security Scheme

Heejin Jang**, Sangwook Kim**

요 약

여러 호스트로 구성된 단일 관리자 영역 네트워크를 안전하게 사용하기 위해 일관적인 보안관리와 침입자 추적 및 정보공유를 통한 자동대응이 요구된다. 본 논문에서는 보안스킴의 동적 확장을 지원하는 미행 메커니즘을 제시하고 이를 기반으로 설계, 구현된 침입자동대응시스템인 ARTEMIS(Advanced Realtime Emergency Management Identification System)를 소개한다. 미행 메커니즘을 기반으로 설계된 보안관리 시스템은 모든 네트워크 컴포넌트를 같은 보안스킴의 일부로 만드는 것이 가능하므로 정책과 네트워크 보안을 일관적으로 관리할 수 있다. 또한 보안관리를 위한 영역과 시간을 동적으로 확장함으로써 침입에 대한 추적과 자동대응 기능을 향상시킨다.

ABSTRACT

It is necessary to control security management consistently and respond to an intrusion automatically in order to use the network securely in the single administrative domain. This paper presents a Shadowing Mechanism supporting a dynamic extension of security scheme and proposes an ARTEMIS(Advanced Realtime Emergency Management and Intruder Identification System), which is designed and implemented based on the suggested technique. It is possible for security management system developed on the basis of the Shadowing Mechanism to make all network components working under the same security scheme. It enhances the accuracy of intrusion tracing and automatic response through dynamic extension of space and time for security management.

Ⅰ. 서 론

시스템에 대한 공격자는 자동화된 공격도구를 이용하여 아주 빠른 속도로 시스템에 피해를 가할 수 있다. 이러한 피해를 최소화하기 위해 차세대 보안관리 시스템은 인간의 개입없이 자동적으로 대응하는 능력을 필요로 한다. 자동대응은 우선 관리자 영역의 네트워크가 시스템 공격을 탐지하고 공격행위를 추적할 수 있어야 한다. 또한 네트워크 내의 호스트간에 공격행위에 대한 정보교환이 가능하고 디바이스에 대한 동적 재설정을 통해 방어의 강도를 조절할 수 있어야 한다[1].

보안관리를 위한 제품들 중 EMERALD[2], AAFID[3], IPA의 IDA[4], ANDIR[5] 등이 자동대응을 제공하는 보안관리시스템의 예이다. 자동대응을 위해서는 직접 제어할 수 있는 보안관리영역의 범위와 공격행위에 대한 정보교환이 중요한 역할을 한다. 하지만 기존의 보안관리시스템들은 독립 시스템으로서 특정

* 본 연구는 한국소프트웨어진흥원 대학 정보통신 연구센터 육성, 지원사업과 BK21 지역대학육성사업 정보기술인력양성사업단의 일부 지원을 받아 수행되었습니다.
** 경북대학교 컴퓨터과학과({janghj, swkim}@cs.knu.ac.kr)

지역문제만을 해결한다. 이들이 분산적으로 배치되면 보안관리를 위한 영역이 넓어지지만 시스템을 설치할 호스트를 결정해야하고 직접 시스템을 설치해야하는 과부하가 따른다. 또한 관리자 영역의 네트워크를 일관적으로 제어하기 위한 정책, 보안관리 등의 유지비용이 많이 든다. AARID와 IDA는 이동 에이전트를 이용하여 보안관리를 위한 영역을 확장하며 자동대응을 제공한다.그러나 이동 에이전트를 위한 플랫폼이 제한된 호스트에 미리 설치되어 있어야 한다는 면에서 지역적인 한계를 극복할 수 없고 이것은 침입자가 여러 네트워크를 거쳐서 침입한 경우 근원지를 밝히기 어렵게 한다. Network Associates 사의 ANDIR은 네트워크 컴포넌트간의 공유 프로토콜인 IDIP[6]와 개별 보안시스템에 탑재되는 에이전트를 이용하여 다양한 이기종 보안시스템을 통합하여 네트워크를 관리할 수 있도록 개발하였지만 실제 자사의 보안 솔루션 사이에서의 연동이라는 한계를 가진다.

단일 관리자 네트워크에 대한 더 이상의 침입을 근본적으로 막기 위해 침입자 추적, 공격 시작 호스트에 대한 대응공격, 타겟 호스트에 대한 방어공격 등의 적극적인 대응이 필요하다. 동일한 보안관리 시스템을 미리 모든 호스트에 설치하는 것은 과부하가 크므로 보안관리 영역의 확장과 정보의 공유를 위해 요구에 따라 단일 보안관리 시킴을 동적으로 확장하는 메커니즘이 요구된다.

본 논문에서는 보안스킴의 자동확장성을 지원하기 위한 미행 메커니즘을 제시한다. 이 메커니즘은 모니터링, 복제, 자기보호기능을 기반으로 침입자에 대한 행위정보와 이동경로를 수집하고 메시지 전송에 의해 정보를 공유한다. 이로써 보안관리를 위한 시간과 영역을 동적으로 확장하며 침입자를 추적하여 신분을 확인하고 타겟 호스트 뿐만 아니라 공격을 시작한 호스트에 대해 적절한 대응도 가능하다. 단일 보안스킴이 확장되므로 네트워크에 대한 일관적인 정책 및 보안관리를 지원한다.

제II장에서 미행 메커니즘에 대해 설명하고 제III장에서는 미행 메커니즘에 의해 보안관리를 위한 영역이 동적으로 확장됨을 보인다. 제IV장에서는 미행 메커니즘을 기반으로 설계, 구현된 침입자동대응시스템인 ARTEMIS[7]의 구조를 소개하고 보안관리를 위한 자동확장 과정을 구현 예와 함께 설명한다. 또한 공격범위가 확장에 따라 탐지 및 대응률에 대한 성능평가를 소개하고 제V장에서 결론을 맺는다.

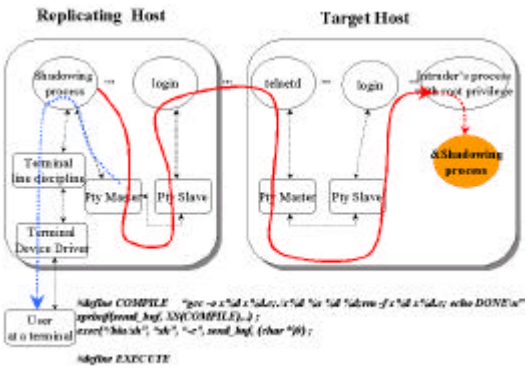
II. 미행 메커니즘

미행 메커니즘은 침입자로 추정되는 사용자의 행위와 사용자의 파일시스템, 계정, 로그인 정보 등을 모니터링한다. 모니터링 정보 중 특히 이동정보와 함께 신분정보가 발견되는 경우 복제과정을 통해 침입자의 타겟 호스트로 침입자가 획득한 권한으로 함께 이동한다. 타겟 호스트에서의 보안관리 작업이 시작됨과 동시에 이전 호스트와의 통신을 통해 보안관리를 위한 정보공유가 가능하다. 이로써 침입자의 이동경로를 따라 보안관리 영역을 확대할 수 있다. 이러한 미행 메커니즘의 과정은 침입자로부터 보안관리행위 자체를 보호하면서 이루어지므로 보안관리를 위한 시간의 확보도 지원한다. 미행 메커니즘을 위한 네트워크 환경은 단일 관리자 영역 또는 서로 협동하는 관리자 영역을 가정하고 이를 신뢰영역이라고 한다. 미행 메커니즘은 침입자의 이동경로와 신분정보를 획득하기 위한 모니터링과 필터링, 침입자 추적을 위한 복제, 호스트간의 정보공유를 위한 메시지 전송, 그리고 이 모든 과정을 보호하기 위한 자기보호로 구성된다.

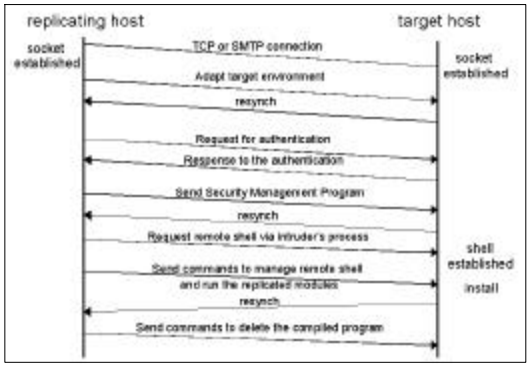
2.1 모니터링과 필터링

네트워크로 연결된 여러 호스트와 시스템에 대한 사용자가 보안관리의 대상이 된다. 모니터링 정보는 호스트 모니터링 정보와 사용자 모니터링 정보로 구성된다. 미행 메커니즘에서는 침입자로 추정되는 사용자가 주된 관심의 대상이 되므로 사용자 모니터링 정보만을 고려한다. 모니터링 결과 획득되는 정보는 그 성질에 따라 동적 모니터링 정보와 정적 모니터링 정보로 나뉜다. 정적 모니터링 정보는 호스트에서의 사용자 디렉토리 및 파일 정보, 사용자가 설치한 백도어, 로그 등으로 구성된다. 사용자가 수행하는 명령, 그에 의해 발생하는 이벤트와 같이 계속적으로 변화하고 실시간 감시가 요구되는 것을 동적 모니터링 정보라 한다. 미행 메커니즘에서 모니터링은 기본적으로 사용자 단위로 수행되며 관리자의 정책에 의해 유용한 정보가 필터된다. 특히 침입자의 이동경로를 파악하고 타겟 호스트로 보안관리를 위한 모듈을 복제하기 위해 이동경로와 신분정보를 걸러내는 필터링 규칙이 필요하다. 이들 정보는 침입자의 터미널을 감시함으로써 획득할 수 있다.

[그림 1]은 미행 메커니즘에서의 모니터링과 복제



[그림 1] 모니터링과 복제



[그림 2] 복제 프로토콜

과정을 나타낸다. 침입자가 공격을 시도하였을 때 가상 터미널을 할당함으로써 침입자의 행위를 감시할 수 있을 뿐 아니라 관리자가 침입자의 프로세스로 명령을 보내는 것이 가능하다. 가상 터미널을 통해 침입자를 감시하면서 telnet, ftp 등의 이동명령 또는 침입자의 ID, 패스워드 등의 신분정보가 발견되면 즉시 복제 프로토콜을 구동하여 보안관리를 위한 모듈의 복사를 실행한다.

2.2 복제 프로토콜

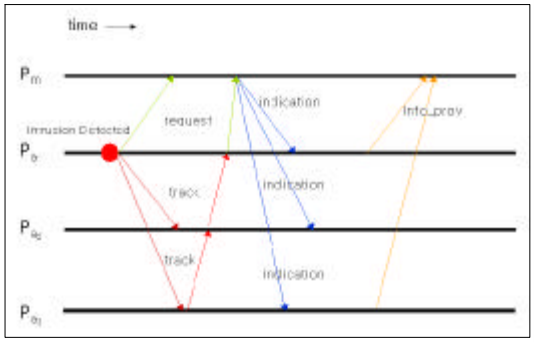
보안스킴의 공간적인 확장성을 지원하기 위해 보안관리 시스템을 타겟 호스트로 복사, 실행하는 과정이 요구된다. 이는 복제 프로토콜[8]을 이용하여 수행되는데 침입자가 다른 호스트, 즉 타겟 호스트를 공격하여 이동하는 경우 미행 메커니즘은 침입자가 획득한 권한으로 타겟 호스트에의 접근이 가능하다. 타겟 호스트의 이전 상태를 보존한 상태에서 보안관리를 위한 모듈을 복사하고 타겟 호스트에서의 보안관리 수행을 위해 이를 실행한다.

우선 보안관리 영역을 확장할 수 있도록 보안스킴을 이동할 시점과 호스트를 선택해야 한다. 침입자가 telnet, ftp, r 명령 등을 이용하여 타겟 호스트로 이동하는 경우 또는 취약점 스캔 후 타겟을 공격하는 프로그램을 실행하여 ID, 패스워드의 신분 정보를 획득하는 시점에서 이동할 타겟 호스트가 설정되고 보안스킴의 이동이 시작된다. 침입자의 쉘을 이용하여 보안관리 프로그램을 복제하므로 침입자가 인식하지 못하도록 타겟 호스트에서의 침입자의 사용환경을 그대로 유지할 필요가 있다. 타겟 호스트에서 이미 보안관리 프로그램이 동작 중인지 확인한다. 이미 설치된 프로그램이 있다면 인증을 통해 정당한 보안관리 프로그램임을 확인한 후 타겟 호스트로의 확장을 종료하고 설치된 프로그램이 없다면 침입자 권한의 쉘을 이용하여 보안관리를 위한 프로그램을 전송하고 컴파일 후 데몬의 형태로 백 그라운드로 실행한다. 이로써 침입자에 대한 추적이 계속된다. 타겟 호스트가 전송하는 resynch 메시지는 타겟 호스트와의 동기화를 위해 사용된다. resynch 메시지를 보냄으로써 복제가 완료되었음을 알리면 복제하는 호스트는 타겟 호스트에서의 모니터링 상태를 보호하기 위해 소스 프로그램을 삭제하고 자기보호를 수행하는 명령을 타겟 호스트에 전송한다. 복제를 수행하는 호스트와 타겟 호스트간의 복제 프로토콜은 [그림 2]와 같다.

2.3 정보공유를 위한 프로토콜

보안관리의 효율성을 높이기 위해 네트워크 내 시스템 간의 정보공유가 요구된다. 정보공유의 일반적인 방법인 복제를 이용하면 데이터 동기화와 일관성 유지 등의 문제를 가지게 되므로 본 메커니즘에서는 메시지 전송을 통해 정보를 공유한다.

타겟 호스트로의 보안스킴 확장이 완료된 후 미행이 시작된 마스터 시스템 P_m과 복제가 수행된 에이전트 시스템 P_{a_1}, P_{a_2}, P_{a_3}간의 메시지 전송을 통해 정보를 전달한다. 에이전트 시스템들은 서로 인접해 있다. 추적(track), 요청(request), 정보제공(info_prov), 지시(indication)와 같은 네 가지 메시지가 사용된다. 추적 메시지는 하나의 에이전트 시스템에서 시스템과 네트워크에 피해를 주는 공격이 될 수 있는 이벤트 또는 정보가 발견되는 경우 주위의 에이전트 시스템에 전송되어 이 공격이 각 에이전트를 거쳤는지 확인하는데 사용된다. 추적 메시지는



[그림 3] 정보공유를 위한 프로토콜

공격으로 의심되는 이벤트가 발생한 호스트, 그 호스트 주위의 에이전트 시스템이 설치된 호스트, 발생한 이벤트의 종류와 공격하기 위해 사용한 연결의 종류로 구성되고 이들이 주위의 호스트에 전송된다. 메시지를 받은 에이전트 시스템은 그 공격과 관련된 이벤트가 자신의 시스템에서 발견되는 경우, 추적 메시지에 정보를 덧붙여 주위 에이전트 시스템으로 전송한다. 추적 메시지의 복사본으로 마스터 시스템 P_m에 전달되는 요청 메시지는 추적 메시지의 구성과 같다. 정보제공 메시지는 마스터 시스템의 요청에 의해 또는 주기적으로 각 에이전트 시스템에서 마스터 시스템으로 전송되어 정보를 제공한다. 지시 메시지를 통해 각 에이전트가 취해야 할 적절한 대응을 지시한다.

[그림 3]은 정보공유를 위한 마스터, 에이전트 시스템간의 일반적인 메시지 교환을 나타낸다. 각 메시지의 구성은 다음과 같다.

메시지 track = {S, DL, E, C, AR}
메시지 request = {S, DL, E, C, AR}
메시지 info_prov = {S, DL, AD}
메시지 indication = {S, DL, AL}
S : Source, DL : Destination Lists,
AL : Action lists, E : Type of Event,
C : Type of Connection,
AR : Appending Records.
AD : Attack relevant Data

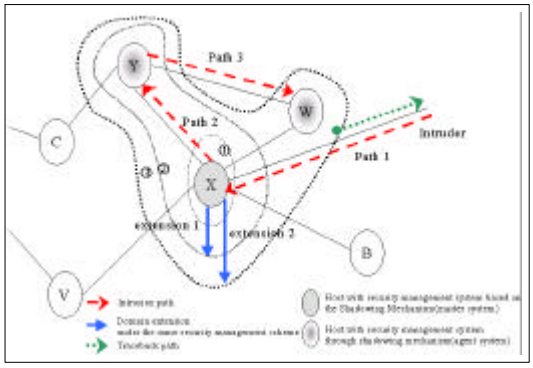
2.4 자기보호

미행 메커니즘에 있어서 자기 보호는 침입자에 대한 모니터링 행위 자체를 보호하는 것이다. 이는 침입자를 감시할 수 있는 시간을 확보하는데 중요한 역할을 한다. 미행 메커니즘에서의 자기 보호는 흔적 지우기와 위장과 같은 두 가지로 분류된다. 흔적 지우기에는 다음과 같은 방법이 사용된다. 프로세스 상태 명령이 프로그램에 의해 호출된 방법을 보여주지 않기 위해 인자를 처리한 후 인자리스트를 지운다. 실행 바이너리를 삭제함으로써 링크 연결이 삭제되어 단지 프로그램 실행에 의해서만 참조된다. 또한 프로그램상의 버그가 흔적 남기는 것을 방지하기 위해 자원사용제한 함수를 사용하여 코어 덤프(core dump)를 막는다. /var/adm/utmp(x) 파일은 현재 시스템에 로그온 중인 모든 사용자 기록을 나타낸다. 이들 기록으로 침입자가 모니터링 행위를 알아차릴 수 있으므로 이 기록을 삭제한다. 자기 보호의 또 다른 방법인 자신을 숨기기 위한 위장을 위해 다음과 같은 방법이 사용된다. 침입자를 모니터링 중인 프로세스를 가장 흔한 프로세스인 쉘로 위장함으로써 모니터링 행위를 보호한다. 프로세스를 포크(fork)하여 부모와 자식으로 만든 후 부모는 빠져나오고 자식은 그대로 남아 수행함으로써 프로세스 재생성의 효과를 가져온다. 그 외의 부가적인 방법으로 트로이 목마와 같이 원래 프로그램을 바꿈으로써 자신을 숨길 수 있다. 프로그램 복제 시 프로세스를 숨기기 위한 ps, top, pidof 프로그램 또는 파일을 숨기기 위해 find, ls, du 프로그램을 함께 보내어 사용이 가능하다.

III. 보안스킴의 자동확장

본 장에서는 미행 메커니즘을 이용하여 보안스킴이 동적으로 확장됨을 보인다. [그림 4]는 하나의 신뢰영역에서 미행 메커니즘에 의한 보안관리 영역의 확장을 보인다. 신뢰영역 내에 미행 메커니즘을 기반으로 보안관리를 수행하는 시스템을 가진 호스트로서 유일하게 X가 있고 외부에서 침입자가 X 호스트를 공격하여 신뢰영역내로 침입하였다. 침입자에 의해 호스트 X에서 Y로 그리고 W로의 공격이 계속된다. 초기에 유일한 보안관리시스템이 제어하는 영역은 ① 이다. path 2를 따라 공격이 수행되면 미행 메커니즘에 의해 X의 보안관리시스템의 일부가 Y로 복제되어 보안관리 영역은 ②로 확장된다. path 3의 공격에 의해 보안관리 영역은 ③으로 확장된다.

침입자가 신뢰영역으로 처음 침입하였을 때 접근한 호스트에만 미행 메커니즘 기반의 보안스킴이 이미 존재하였다고 가정한다. 신뢰영역 내에서의 침입



[그림 4] 미행 메커니즘에 의한 보안관리 영역의 확장

자가 수행한 모든 이벤트는 침입자의 이동경로상의 각 호스트에서 침입자에 대해 수집되는 정보와 같다는 것을 보인다. 이를 위해 고려할 정보는 침입자에 대한 동적 모니터링 정보이다.

모니터링 정보 M은 정적 모니터링 정보 SM과 동적 모니터링 정보 DM로 구성되고 모니터링의 결과는 상태와 상태전이 그리고 전이를 유발한 이벤트들을 기록한 데이터이다.

[정의 1] 정적 모니터링 정보는 상태 모니터링 함수에 의해 수집된다. 상태 모니터링 함수 σ_{state}는 $\sigma_{state} \rightarrow N_E \times N_V$이고 모니터링 대상의 연관된 상태 σ를 모니터링 대상의 이름 N_E와 그것의 값 N_V로 기록한다. 정적모니터링정보 SM은 $SM = N_E \times N_V$와 같이 표현한다.■

정적 모니터링 정보는 각 사용자와 관계가 있으면서 시스템 상태와 연관된 컴포넌트를 기록한다.

[정의 2] 동적 모니터링 정보는 변화 모니터링 함수에 의해 수집된다. 변화 모니터링 함수 σ_{change}는 $C_\sigma \times \sigma \rightarrow N_C \times N_V \times N_E$이고 모니터링 대상의 상태를 변화시킨 이벤트 C_σ,와 상태 σ를 이벤트의 이름 N_C, 모니터링 대상의 이름 N_E와 그것의 값 N_V로 기록한다. 동적 모니터링 정보 DM은 $DM = N_C \times N_V \times N_E$와 같이 표현한다.■

동적 모니터링 정보는 시스템 상태와 연관된 컴포넌트를 변경시키고 또한 그러한 컴포넌트의 값을 변경시키는 특정 이벤트를 기록한다. 예를 들어 시스템 호출이 파일에 대한 사용자의 권한을 변경할 때마다 그 모든 것을 기록한다고 가정한다. N_C는 시스템 호출의 이름이고 N_E는 파일과 사용자들의 이름이며 N_V는 보호 모드의 새로운 설정이다. 동적 모니터링 정보에서 이벤트는 단일 프로세스에 의해 내부적으로 발생하는 이벤트와 외부적으로 관찰될 수 있는 이벤트로 나누어진다. 프로세스에 의해 수행되는 함수호출이 내부적으로 발생하는 이벤트의 예이고 외부적으로 관찰될 수 있는 이벤트는 콘솔상에서의 사용자가 수행하는 명령을 포함한다. 발생하는 이벤트는 시간적 순서와 공간적 위치를 가진다.

[정의 3] dm은 단일 동적 모니터링 정보를 나타내고 DM은 $\{dm_1, dm_2, dm_3, \cdots, dm_i, \cdots\}$로 구성된다. dm은 (c, e, v)의 쌍으로 구성되고 c는 발생한 이벤트, e는 이벤트가 발생한 객체, v는 변경된 객체의 값을 나타낸다. 각 단일 모니터링 정보는 $t(dm_i)$라는 이벤트 발생시각을 가진다. 이러한 단일 모니터링 정보들은 전체적으로 순서화되어 있으므로 $i \geq 1$인 모든 i에 대해 $t(dm_i) \leq t(dm_{i+1})$이다.■

[정의 4] 모니터링 정보 M을 구성하는 정보 중에서 호스트 H_1상에서의 모니터링 정보 M^{H_1}과 호스트 H_2상에서의 모니터링 정보 M^{H_2}가 주어질 때, 이러한 두 호스트 상에서의 모니터링 정보를 통합한 결과는 $M^{H_1} \oplus M^{H_2}$으로 나타내고 이는 M의 부분정보이다.■

정의 4는 동적 모니터링 정보, 정적 모니터링 정보 각각에 모두 적용된다.

[정의 5] 필터함수 F_p는 모니터링 정보 M을 M의 부분 정보인 σM으로 매핑하는 함수이다. σM은 M에서 유용하지 않은 정보들을 삭제한 결과이다. p는 수집된 정보 중 필요없는 정보들을 삭제하기 위한 필터링 규칙이다.■

F_{p_i}는 침입자 이동경로 및 신분정보 필터 함수이고 p_i는 수집된 정보 중 이동경로와 신분정보를 걸러내기 위한 필터링 규칙이다. 이러한 필터링 결과 수

집되는 모니터링 정보 M는 침입자가 이동한 호스트, 호스트 접근시 사용한 ID, 패스워드 등의 신분 정보를 가진다. F_{p_a}는 침입자 행위정보 필터함수이고 p_a는 수집된 정보 중 행위정보를 걸러내기 위한 필터링 규칙이다. 모니터링 정보 M는 침입자가 호스트의 콘솔 상에서 수행한 명령뿐만 아니라 명령수행을 위한 프로세스, 그에 의한 함수호출 등의 정보를 가진다.

[정리 1] DM_N $(1 \le N \le k)$을 사용자 N에 대한 동적 모니터링 정보로 정의하면 사용자에 A 대한 동적 모니터링 정보 DM_A를 $DM_A = DM_A^{H_1} \oplus DM_A^{H_2} \oplus \ldots \oplus DM_A^{H_n}$으로 나타낼 수 있다.■

사용자 A에 의해 서로 다른 공간적 위치에서 발생하는 이벤트, 객체, 값의 쌍의 집합 E_A는 $E_A = \{dm_1, dm_2, \ldots, dm_n\}$, n≥1로 나타낼 수 있다. 사용자 A가 호스트 H_1에서 $DM_A^{H_1} = dm_{H_11}, dm_{H_12}, dm_{H_13}, \ldots, dm_{H_1k}$과 같은 이벤트, 객체, 값의 쌍들을 생성하고 다시 호스트 H_2로 옮겨 $DM_A^{H_n} = dm_{H_21}, dm_{H_22}, dm_{H_23}, \ldots, dm_{H_2p}$를 결국 H_n으로 옮겨 $DM_A^{H_n} = dm_{H_n1}, dm_{H_n2}, dm_{H_n3}, \ldots, dm_{H_nq}$를 생성하였다고 가정한다. 또한 $H_11, H_12, \ldots, H_1k, H_21, \ldots, H_2p, \ldots H_n1, \ldots, H_nq$이 $1, 2, \ldots, k+p+q$의 부분순서열이라고 가정한다. 미행 메커니즘은 사용자의 이동경로를 따라 A가 수행하는 이벤트들을 모두 모니터링할 수 있으므로 결과적으로 생성되는 M_A는 $DM_A = DM_A^{H_1} \oplus DM_A^{H_2} \oplus \ldots \oplus DM_A^{H_n} = dm_1, dm_2, \ldots, dm_{k+p+q} = E_A$이다. 침입경로상의 모든 호스트에서의 사용자 A의 행위는 미행 메커니즘에 의해 사용자 단위로 각 호스트에서 수집한 동적 모니터링 정보의 합집합과 같다. 그러므로 단일 관리자 영역의 네트워크 내의 하나의 호스트에 침입자가 공격을 하였고 그 호스트에 미행 메커니즘 기반의 보안관리 스킴이 존재한다면 보안관리 결과는 네트워크 내의 침입자 공격경로상의 모든 호스트에 보안관리를 위한 시스템을 설치하고 수행한 결과와 같다.

Ⅳ. 침입자동대응에의 적용

본 장에서는 제안된 미행 메커니즘을 기반으로 설계, 구현된 침입자동대응시스템인 ARTEMIS를 소개한다. 대응의 한 방법으로써 침입자 추적을 수행하기 위해 영역을 동적으로 확장하는 과정을 보이고 탐지 및 대응률에 대한 성능평가 결과를 보인다.

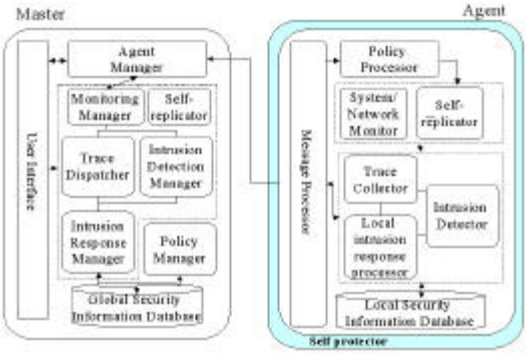
4.1 ARTEMIS : 침입자동대응시스템

ARTEMIS 는 보안관련 정보수집을 위한 모니터링, 침입탐지, 침입대응을 인간의 개입없이 자동적으로 수행하는 침입자동대응시스템이다. 본 시스템은 하나의 마스터 시스템과 다수의 에이전트 시스템으로 구성된다. 즉, 미행 메커니즘을 기반으로 하므로 필요한 곳에 에이전트 시스템을 동적으로 설치하는 것이 가능하다. [그림 5]는 ARTEMIS의 전체 구조를 나타낸다.

에이전트 시스템과 마스터 시스템은 모두 모니터링, 침입탐지, 침입대응 등의 보안관리를 위해 복제를 수행한다. 에이전트는 에이전트가 존재하는 각 호스트에서의 지역적인 정보만을 수집하고 그에 따라 지역적인 침입증상만을 탐지하여 대응을 수행한다. 마스터는 에이전트 관리기(Agent Manager)를 통해 네트워크에 분산된 모든 에이전트들을 관리한다.

자기보호 및 복제 마스터 시스템과 에이전트 시스템은 특정 조건이 만족되는 경우 모니터링, 침입탐지, 침입대응을 위한 모듈을 자기복제기(Self Replicator)를 통해 다른 호스트로 전송, 실행하여 자신을 복제한다. 이로써 보안관리를 위한 영역을 넓힐 수 있다. 또한 모니터링, 침입탐지, 대응을 위한 모든 모듈들은 자기보호기능을 제공하는 자기보호기(Self Protector)의 지원을 받아 보안관리를 위한 시간을 확장할 수도 있다.

모니터링 각 에이전트 시스템의 시스템/네트워크 모니터(System/Network Monitor)는 보안관련



[그림 5] ARTEMIS 구조

정보를 수집한다. 보안관련 정보는 2.1절에서 설명한 바와 같이 동적 정보와 정적 정보를 포함한다. 이벤트는 단일 프로세스에 의해 내부적으로 발생하는 이벤트와 외부적으로 관찰될 수 있는 이벤트로 나누어진다. 프로세스에 의해 수행되는 함수호출이 내부적으로 발생하는 이벤트의 예이다. 반면 외부적으로 관찰될 수 있는 이벤트는 사용자가 수행하는 명령어들을 포함한다. 에이전트에서 수집된 정보들은 마스터 시스템의 요청에 의해 또는 정기적으로 마스터로 전송되어 모니터링 관리기(Monitoring Manager)에서 관리된다.

침입탐지 에이전트 시스템의 침입 탐지기(Intrusion Detector)는 모니터링 결과 수집된 정보로부터 의미적인 정보를 추출하여 지역적인 침입증상을 탐지한다. 마스터 시스템의 침입탐지 관리기(Intrusion Detection Manager)는 에이전트로부터 전송되는 보안정보들을 분석하여 관리하는 네트워크 상의 침입을 탐지한다. 침입판정은 프로세스가 시스템 콜을 호출하는 시점에서의 프로세스의 권한 상태 즉, 프로세스의 UID, EUID, GID, EGID를 이용하여 유한상태기계(Finite State Machine) 모델을 구축하고 상태전이에 의해 침입을 탐지한다. 또한 각 사용자의 터미널을 감시하여 수행되는 사용자 명령어들을 파싱하여 얻은 분석 결과와 시스템 호출의 부가적인 정보를 종합적으로 이용하여 오용탐지방법의 문제인 오류탐지를 해결한다.

침입대응 신속한 대응이 요구되는 경우 지역침입대응처리기(Local Intrusion Response Processor)가 미리 설정된 정책에 따라 적절한 대응을 취한다. 에이전트 시스템에서 수행되는 침입대응에 대한 모든 정보는 마스터 시스템에게 전달된다. 마스터 시스템은 각 에이전트로부터 전달된 보안정보, 침입증상, 침입대응 등을 분석하여 침입대응 관리기(Intrusion Response Manager)를 통해 전역적 차원에서의 대응을 수행한다. 마스터 시스템이 직접 대응을 수행할 수도 있지만 각 에이전트에 적절한 대응을 지시할 수도 있다. 적절한 대응을 전달받은 에이전트는 지역침입대응처리기를 통해 침입에 대한 대응을 수행한다. 단순한 보안관리현황 디스플레이, 이메일 또는 콘솔 메시지를 이용한 알림뿐만 아니라 정책설정 변경, 침입자 추적, 타겟 호스트, 공격을 시작한 호스트에 대한 제어 및 동적 설정 변경 등이 ARTEMIS에서 제공하는 대응의 예이다.

마스터와 에이전트간의 정보공유 네트워크 상의 침입추적과 탐지, 대응 등의 보안관리를 위해서는 호스트들간의 정보공유가 중요한 역할을 한다. 2.3 절에서 소개한 메시지 전송기법을 사용하여 ARTEMIS는 마스터 시스템과 에이전트 시스템간에 정보를 전달한다. 요청 메시지는 각 에이전트에 존재하는 추적 수집기(Trace Collector)로부터 마스터에 있는 추적 실행기(Trace Dispatcher)로 전송된다. 이 메시지는 추적 실행기에 의해 공격의 이동경로 분석, 침입 추적에 사용된다.

정책설정 마스터 시스템의 정책 관리기(Policy Manager)는 관리자에 의해 초기화된 정책을 유지하면서 침입대응의 결과 재설정되는 정책을 반영하여 네트워크 전역에 적용되는 정책과 각 에이전트에 적용되는 지역정책을 변경을 지시한다. 에이전트 시스템의 정책 처리기(Policy Processor)는 지역 정책을 직접 수행, 변경한다.

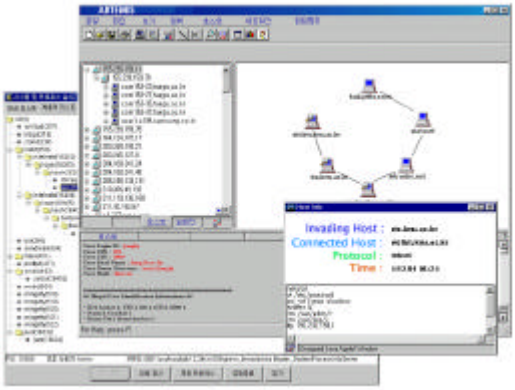
ARTEMIS 사용자는 초기 정책 설정, 관리 영역에 의한 에이전트 수동 설치 등을 입력할 수 있다. ARTEMIS가 설치된 호스트와 침입자의 이동 경로상에 있는 호스트에 대한 침입탐지, 침입추적 등을 수행하고 각 호스트에 대한 모니터링 정보, 침입추적 결과, 침입자의 이동경로상의 호스트에 대한 정보, 침입자에 대한 정보를 보고서의 형태로 관리자에게 제공할 수 있다. ARTEMIS는 시스템간의 인터페이스 및 사용자 인터페이스의 구현을 위해 J2EE (Java 2 Platform, Enterprise Edition)를, 그 외 모니터링, 침입탐지, 대응을 위한 모듈의 구현을 위해 GNU C/C++ 2.7.x.x를 이용한다. 감사 데이터 및 모니터링 정보를 저장하는 DBMS로는 MySQL 3.22.x를 사용하고 시스템 간의 인증 및 암호를 위해서 JCE(Java Cryptography Enhancement)1.2 패키지를 사용한다.

4.2 침입자 추적을 위한 동적 확장

ARTEMIS는 하나의 호스트에 마스터 시스템을 인스톨하면서 초기화된다. 마스터 시스템을 설치하면서 단일 관리자 영역에 있는 모든 호스트에 에이전트 시스템을 설치하는 것이 가능하지만 이는 시스템을 인스톨, 유지하는데 부가적인 비용을 필요로 한다. 그러므로 침입자가 마스터 시스템을 거쳐 다른 호스트로 이동하는 경우 그 호스트가 단일 관리자 영역에 존재한다면 에이전트 시스템은 미행 메커니즘의 복제과정을 통해 자동으로 타겟 호스트에 설

치된다. 에이전트 시스템들은 침입자의 이동경로를 따라 필요에 의해 계속적으로 설치될 수 있다. 네트워크 상에서 침입이 탐지되는 경우 각 에이전트에 미리 설정된 정책에 의해 침입자 추적, 타겟 호스트 상에서의 대응, 침입을 시작한 호스트에 대한 대응, 설정 변경, 보고서 제공 등의 대응이 자동적으로 이루어질 수도 있고 마스터 시스템에 의해 전역적인 대응이 이루어지기도 한다. [그림 4]는 침입자 추적을 위한 보안스킴의 동적 확장과정을 보인다. 침입자가 여러 네트워크 컴포넌트를 거쳐 공격을 시도하는데 마스터 시스템을 거치는 경우 신뢰영역 내에서는 이동경로를 따라 에이전트 시스템을 동적으로 설치하여 침입자를 미행한다. 도중 발생하는 행위정보와 이동정보, 타겟 호스트 공격을 위해 침입자에 의해 사용되는 사용자의 파일시스템을 조사, 분석하여 공격을 시작한 호스트를 추적한다.

[그림 6]은 ARTEMIS에 의해 수행된 보안관리 결과, 침입자로 추정되는 사용자에 대한 정보와 침입자가 이동한 경로를 보여준다. 침입자가 마스터 시스템이 설치된 master.knu.ac.kr에 대해 침입을 시도하고 그 호스트를 거점으로 하여 다른 호스트들을 공격할 경우 ARTEMIS는 침입자의 이동경로를 따라 via.knu.ac.kr와 victim.knu.ac.kr에 에이전트 시스템을 자동으로 설치한다. ARTEMIS가 설치된 호스트에서 침입자가 백도어 프로그램의 설치 등을 목적으로 이전 침입경로 상에 있는 호스트에 접속하거나 침입을 시작한 호스트에 접속하는 경우 침입자의 신분과 침입을 시작한 호스트를 확인할 수 있다. ARTEMIS가 설치된 호스트들은 서로 간의 정보공유가 가능하므로 각 에이전트 시스템에서 수집된 정보가 마스터 시스템에서 전송, 분석되어 원래 ARTEMIS가 존재하지는 않지만 확장가능한, 제한된 네트워크 상에서의 탐지, 추적 및 대응이 가능하다. [그림 6]에서 master.knu.ac.kr와 via.knu.ac.kr을 거쳐 victim.knu.ac.kr 호스트에 대한 공격을 시도한 호스트는 kail.pitts.com 이라는 것을 알 수 있다. [그림 6] 하단의 프레임은 via.knu.ac.kr에서 victim.knu.ac.kr을 공격하는 데 사용하는 침입자의 ID 정보를 나타내고 하단의 창은 침입자의 행위정보를 실시간으로 보인다.



[그림 6] ARTEMIS에서 침입자 추적

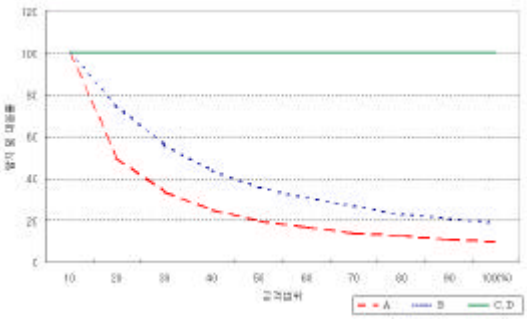
4.3 성능평가

시스템에 대한 공격은 간단하게 타겟 시스템에 대한 정보수집, 타겟 시스템으로의 침입, 재침입을 위한 백도어 설치의 세 단계로 이루어진다[9]. nmap등의 스캔도구를 이용하여 이동할 호스트에 대한 정보를 수집한 후 타겟 시스템으로의 공격을 수행한다. 제안한 시스템의 성능을 평가하기 위해 [표 1]의 조건으로 리눅스 커널 2.2의 sendmail[10] 취약점 공격을 예로 대응의 효율성과 비용에 대하여 살펴본다. BUGTRAQ[11]에 발표된 sendmail 취약점은 리눅스 커널의 취약점이다. 이를 이용하면 비특권 프로그램으로써 특권 프로그램인 sendmail이 특권을 낮추지 못하도록 하여 타겟 호스트상에서의 루트 권한을 획득할 수 있다. 성능평가를 위한 환경은 리눅스 커널 2.2 이상의 운영체제를 가진 호스트들로 구성된 서로 신뢰하는 영역이다.

평가의 조건은 다음과 같다. 공격자가 여러 호스트들로 구성된 신뢰영역의 하나의 호스트에 침입하고 모든 침입은 sendmail 취약점을 이용한다. [표 1]의 D의 경우는 침입자 탐지와 함께 미행을 수행하는 ARTEMIS를 설치하고 A, B, C의 경우는 보안스킴의 확장이 적용되지 못하도록 미행 메커니즘이

[표 1] 평가 조건

조건 / 경우	침입탐지 및 대응 시스템 설치 여부	이동성 여부
A	침입경로상의 하나의 호스트에 설치	제공안됨
B	침입경로상의 두 개의 호스트에 침입탐지 및 대응 시스템 설치	제공안됨
C	모든 호스트에 미리 설치	제공안됨
D	침입경로상의 하나의 호스트에 설치	침입자 이동경로에 따라 제공



[그림 7] 공격범위 확장에 대한 평균 탐지 및 대응률

배제된 침입탐지 및 대응시스템을 설치한다. [그림 7]은 전체 신뢰영역을 100이라고 했을 때 침입자에 의한 공격범위가 확장됨에 따라 공격경로상의 각 호스트에서의 평균 대응률에 대한 실험결과를 나타낸다. 신뢰영역으로의 침입의 경우 항상 처음 공격을 시도하는 호스트에 침입탐지 및 대응 시스템이 설치되어 있다고 가정한다. [그림 10]에서 긴 점선으로 표시된 A 경우는 신뢰영역에 대한 공격을 시도하는 첫 번째 호스트에만 탐지 및 대응 시스템이 존재하고 침입자가 신뢰영역에 대한 공격범위를 넓혀감에 따라 평균 탐지 및 대응률이 급격히 떨어진다는 것을 보인다. 짧은 점선으로 표시된 B 경우는 공격을 시도하는 첫 번째 호스트와 공격경로 상의 임의의 위치의 하나의 호스트에 침입탐지 및 대응을 위한 시스템이 존재하고 이는 A 경우보다는 평균 대응률이 낮지만 공격범위가 넓어짐에 따라 여전히 대응률이 떨어진다는 것을 알 수 있다. C와 D 경우는 실험결과 모든 호스트에서 침입탐지 및 대응이 가능하다. 하지만 모든 호스트에 대해 미리 탐지 및 대응을 위한 시스템을 설치하는 C 경우와 미행 메커니즘에 의해 침입자를 추적하면서 보안스킴을 확장해 가는 D의 경우를 비용면에서 비교하면 설치와 차후의 일관적인 정책, 보안관리가 자동적으로 이루어지는 D가 상당한 장점을 가진다.

V. 결 론

현재 공격자들이 많이 이용하는 분산 서비스 거부 공격의 경우 대부분의 공격자는 자신의 호스트 이외의 다른 호스트를 거점으로 공격의 출발지 주소를 위장하여 타겟 호스트를 공격함으로써 공격자 자신의 위치를 숨긴다. 타겟 호스트가 단독으로 이러한 공격 기술에 대해 대응하는 것은 불가능하다. 그러므로 네트워크 경계를 넘나드는 공격을 자동적으로 추적하고 막기위해 협동하는 기술이 요구된다. 본 논문에서 제시하는 미행 메커니즘은 네트워크 경계를 넘나드는 공격을 자동적으로 추적하고 막기위해 협동하는 기술적 수단을 제공한다. 미행 메커니즘은 모니터링, 복제, 자기보호, 메시지 교환을 통해 서로 신뢰하는 영역 내에서 보안스킴을 자동적으로 확장시킨다. 이로써 단일 보안스킴에 의해 보안관리가 수행되어 넓은 영역을 관리하기 위해 개발되었던 기존 보안 시스템의 문제였던 일관적인 정책과 보안관리가 가능하다. 특히 침입자의 이동경로를 따라 보안스킴이 확장되므로 침입자에 대한 추적도 가능하다. 메시지 교환을 통해 지역적으로 수집하는 보안정보를 공유하여 여러 호스트에 걸친 침입에 대해 다양한 보안정보를 획득할 수 있다. 이를 포괄적으로 분석하여 보다 정확한 침입탐지와 추적, 적절한 대응이 가능하다. 이러한 미행 메커니즘은 플러그 인의 형태로 모니터링, 침입탐지 및 차단, 대응 등의 보안관리를 위한 여러 시스템에 적용될 수 있다.

참 고 문 헌

[1] "Automatic Intrusion Tracing and Response", NAI Labs advanced research, http://www.nailabs.com.

[2] P.A. Porras and P.G. Neumann, "EMERALD : Event Monitoring Enabling Responses to Anomalous Live Disturbance," *Proceedings of the 20th National Information Systems Security Conference*, pp. 353 365, October 1997.

[3] J.Balasubramaniyan, J.O.Garcia-Fernandex, D.Isacoff, E.H.Spafford, and D. Samboni, "An Architecture for Intrusion Detection using Autonomous Agents," Coast TR 98-05, Dept. of Computer Science, Purdue Univ., 1998.

[4] W.Jansan, P.Mell, T. Karygiannis and D. Marks, "Mobile Agents in Intrusion Detection and Response," *Proceedings of the Canadian Information Technology Security Symposium*, June 2000.

[5] M.Asaka, A.Taguchi, and S.Goto, "A

Method of Tracing Intruders by use of mobile Agents," *Proceedings of INET*, June 1999.

[6] D. Schnackenberg and K. Djahandari, "Infrastructure for Intrusion Detection and Response", http://seclab.cs.ucdavis.edu/projects/idip.html

[7] **장희진, 박보석, 김상욱, "미행 메커니즘에 의한 침입자동대응," 한국통신정보보호학회 종합학술발표회논문집**, pp. 514 522, 11, 2000.

[8] H. Jang and S. Kim, "A Self-Extension Monitoring for Security Management," *Proceeding of the 16th Annual Computer Security Applications Conference*, pp. 196 203, December 2000

[9] **"네트워크 공격기법의 파라다임 변화와 대응방안"**, CERTCC-KR-TR-2000-05, 2http://www.certcc.or.kr

[10] Sendmail Consortium. http://www.sendmail.org/, 2000

[11] Wojciech Purczynski, Sendmail & Procmail local root exploits on Linux kernel up to 2.2.16pre5. BUGTRAQ Mailing list (bugtraq@securityfocus.com), 2000. Message ID : <Pine.LNX.4.21.0006090852340.3475300000@alfa.elzlbsoft.pl>

<著者紹介>

장 희 진 (Hee-jin Jang) **학생회원**
1997년 2월 : **경북대학교 컴퓨터과학과 졸업**
1999년 2월 : **경북대학교 컴퓨터과학과 석사**
1999년 3월 현재 : **경북대학교 컴퓨터과학과 박사과정**
<관심분야> **정보보안, 침입탐지 및 대응, 시스템/네트워크 모니터링, 이동 컴퓨팅**

김 상 욱 (Sang-wook Kim) **정회원**
1979년 2월 : **경북대학교 컴퓨터과학과 졸업**
1981년 2월 : **서울대학교 컴퓨터과학과 석사**
1989년 2월 : **서울대학교 컴퓨터과학과 박사**
1988년~ 현재 : **경북대학교 컴퓨터과학과 교수**
<관심분야> **이동 멀티미디어 컴퓨팅, 정보보호**