하부노드 보안관리을 위한 에이전트 설계

배현철° 김상욱°°
°경북대학교 정보보호학과
°°경북대학교 컴퓨터과학과
{hcbae°, swkim°°}@woorisol.knu.ac.kr

Agent Design for Sub-Node Security Management

HyunChul Bae° Wook-Sang Kim°°
°Dept. of Information Security, °°Dept. of Computer Science,
Kyungpook National University

요 약

본 논문에서는 네트워크와 라우터, 방화벽 등 유무선 네트워크를 구성하는 하부노드들의 보안관리를 위한 에이전트을 설계하였다. 또한 제시하는 에이전트는 에이전트 상위에서 정책 언어를 이용하여 다양한 하부 노드로 구성되는 네트워크에 대하여 동적인 관리에 대한 확장성을 지원하며, 해당 하부 노드를 관리하는 에이전트를 구성하여 정책 적용에 있어 효율을 증가시키고, P2P 기술을 적용하여 하부 노드에 문제 발생시 인접한 하부 노드들의 에이전트들 간에 문제를 해결하고 미연에 방지할 수 있도록 한다.

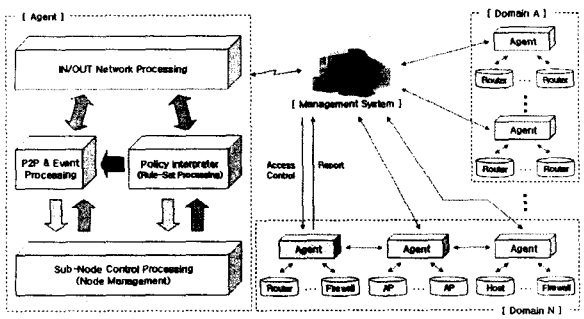
1. 서 론

현재의 네트워크는 구성하는 장비 또는 시스템의 수와 규모가 크며, 정확한 세부 정보를 파악하기 어렵다. 또한, 네트워크 구성이 복잡하고 구성 요소의 변동이 심하며 구성 요소의 종류가 매우 다양하기 때문에 공통적인 방식으로 접근하기가 불가능하다. 때문에 기존의 중소규모의 노드관리 방식과 도구로는 효과적인 결과를 기대할 수 없다. 구성 요소의 다양성으로 직접적인 접근 방식을 사용할 경우 절차적으로도 매우 번거로우며, 시간적으로 매우 비효율적이다. 세부 정보의 결여로 효과적인 제어도 쉽지 않다. 특히 노드에 대한 긴급한 제어가 요구되더라도 다양한 기종에 일일이 접근해야 하는 절차적 복잡성으로 인해 적절한 대응이 어렵다. 때문에, 노드관리를 효율적, 효과적으로 수행하기 위해서는 다양한 구성 요소에 일괄적으로 접근할 수 있는 방법이 필요하다. 노드관리에는 다양한 구성 요소와 이들 사이의 복잡한 관계로 구성되어 있다. 때문에 그것을 관리하고 제어하기 위해서는 자동화된 관리 메커니즘이 요구되며, 그러한 메커니즘에 의한 실제적인 구성 요소에 대한 접근과 제어를 위해서는 일정 수준의 세부 정보와 노드를 관리해 주는 에이전트가 필요하다. 이에 2장에서 에이전트에 언급하며, 3장에서 설계한 에이전트의 동작과정에 대해서 설명하고 4장에서 결론을 맺는다.

2. 에이전트

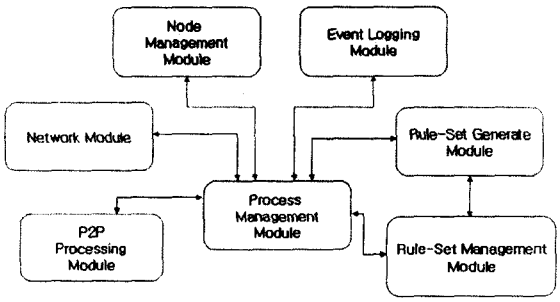
하부 노드에는 여러 종류의 라우터와 방화벽, 호스트 AP 등의 다양한 노드들이 존재한다. 관리자가 설정한 정책을 하부의 여러 노드들은 자신이 속한 도메인에 적용할 수 있어야 하며, 정책의 방향에 맞게 올바르게 수행할 수 있어야 한다.[1]

따라서 올바른 정책을 적용하기 위해서는 하부 노드들마다 상위 계층에서 내려온 정책 및 지침들을 수용할 수 있게끔 자신들만의 형태로 바꾸어 주는 에이전트를 필요로 하며, 하부 노드의 상황에 따라서 문제가 발생한 노드에 대해 문제를 해결하고 인접한 노드들 간에는 문제가 확산되지 않게 미연에 방지하게끔 정책을 설정하는 기능이 필요하다. 즉, 특정 하부 노드에 바이러스나 외부 침입자의 공격으로 인한 트래픽 증가 시에 관리자에게 통보 후에 이를 위한 정책을 적용하는데 까지 시간이 걸리게 되고 문제 해결을 위한 정책 작성 및 작성된 정책의 적용하는 동안 시간 지체 인하여 문제가 확산되어 해당 도메인의 네트워크가 마비되는 문제가 발생할 수 있으므로, 이에 P2P 기술을 적용하여 해당 도메인에 포함되고 인접한 에이전트들 간에 하부 노드의 상태에 따라 스스로 연계하여 이를 처리하는 기능을 필요로 한다. 에이전트의 가장 핵심적인 역할은 상위로부터의 새로운 정책 수신, 상위 정책을 각 하부 노드 타입의 Rule-Set을 생성기능, 각 하부 노드로의 통신 모듈 중 Rule-Set 적용을 위한 Telnet 과 SNMP 을 통한 실제 작업 지시 과정, 적용의 유·무를 판단하는 결과 확인 과정, 정책 결과를 저장하는 부분, 마지막으로 속은 하부 노드와의 상태를 확인하기 위한 주기적인 통신 과정과 인접 하부 노드들의 에이전트들 간에 처리를 위한 P2P 통신 과정으로 이루어진다. <그림 1>은 에이전트의 전체적인 구조를 나타낸 것이다.



<그림 1> 에이전트 전체구조

하위 노드들은 상당히 다양한 종류의 장치들로 구성될 수 있고 그에 따라 적합한 행동을 취하게 하기 위하여 존재하는 것이 각 노드에 맞는 에이전트이다. 각 에이전트는 관리자로부터 전달받은 정책을 자신의 하위 노드에 맞는 Rule-Set을 만들어 내고 그것을 적용하는 것이다. 이러한 하위 노드 관리 시스템은 <그림 2>와 같이 크게 몇 가지 모듈로 나뉜다.



<그림 2> 에이전트 전체 구조

첫째가 Process Management 모듈이다. 이는 에이전트 전체의 스케줄링을 담당하고 특정 루틴의 결과 값을 저장해 두었다가 다른 모듈이 사용할 수 있게끔 한다. 전체적인 프로그램의 동작 흐름을 제어하고 있는 모듈이다.

둘째가 Network 모듈이다. 관리자 혹은 하부 노드와의 모든 데이터 송·수신을 담당한다. 각 하위 노드들에게 직접적으로 정보를 전송하는 부분이다.[5]

셋째가 Rule-Set Generate 모듈이다. 전송 받은 정책을 분석하여 실제 하부 노드에 맞는 Rule-Set을 만들어 낸다. 각각의 하부 노드들에 맞는 형태의 Rule-Set을 만들어 낸다.[2][3]

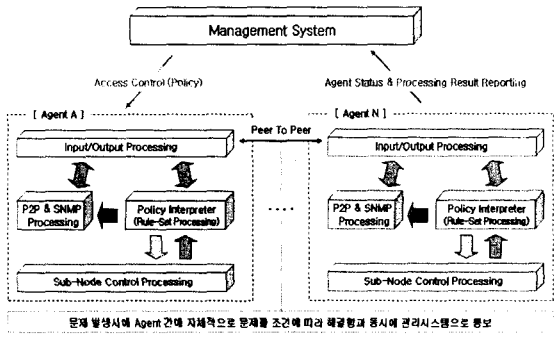
넷째는 Rule-Set Management 모듈이다. Rule-Set Generate 모듈이 만들어낸 Rule-Set이 하부 노드에 적용이 되었다면 그 결과를 보관하게 된다. 또한, 하부노드에 상태에 따른 동적인 Rule-Set을 적용하는 역할을 한다.[4]

다섯째, Node Management 모듈이다. 에이전트 하부에 있는 노드가 어떤 타입인지 알고 있어야 하고, 적절한 주기마다 에이전트와 하부 노드는 통신을 하여 서로의 상태를 감지하고, 이상 유무를 체크하게 된다.

여섯째, P2P Processing 모듈이다. 이는 에이전트들 간에 상태에 따른 자체적인 처리를 주목적으로 하고 있으며, 동일 도메인내의 인접한 에이전트들 간에 정책전송 및 서로간의 상태 확인 등의 처리 기능을 가지고 있으며, 에이전트들 간에 통신을 하는 역할을 가지고 있다.[7][8][9]

마지막으로 Event Logging 모듈이 있다. SNMP를 이용하여 에이전트 하부에 신규로 추가되거나 삭제된 노드들을 확인하게 되며, 해당 하부노드의 세부정인 정보를 파악하는 역할을 한다.[6]

이렇게 7개 module이 모여서 에이전트를 이루고 이러한 에이전트들은 다양한 종류의 하부노드를 관리하게 된며 필요에 따라서 하부 노드에 여러 에이전트가 사용될 수 도 있다. 하부 노드 에이전트는 다양하고 이질적인 하부 노드의 접근과 제어 방식을 통일하여 관리 주체에게 추상적인 네트워크 관리 모델을 제공한다. 많은 수의 노드들을 소수의 에이전트가 관리 가능하도록 하여 물리적인 제약이 최소화할 수 있다. <그림 3>와 같이 리포팅를 통하여 하위 노드에서 적용된 과정이나 오류가 나타난 경우, 동일 도메인내의 에이전트에서 발생된 문제에 대해서 자체적으로 처리한 내용과 문제 발생 내역 등을 관리자가 모니터링 가능하도록 관리자에게 관련 내용을 리포팅하여 유연하면서도 효율적인 관리가 가능하도록 설계되었다.



<그림 3> 에이전트에서 관리자로의 통보

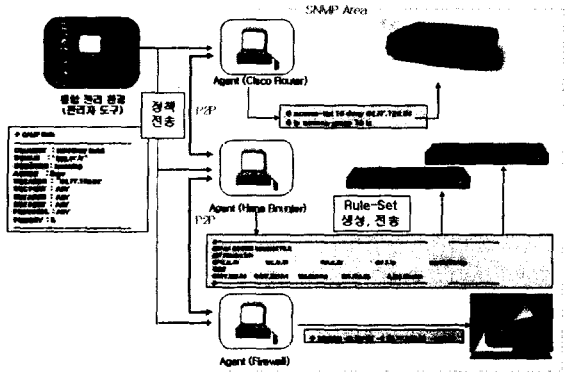
대기 상태, 입력 상태, 인증 과정, Rule-Set 생성 과정, 각 하부 노드들에게 Rule-Set을 전송하는 과정, 결과의 확인 과정, 하부노드 상태에 따른 하부 노드 에이전트 간의 통신을 위한 P2P 처리 과정 중에서 관리자에게서 하부 노드로의 정책이 전달되는 매개체로서 기능과 문제 발생시 서로 간에 협동을 통하여 문제가 전파가 되는 것을 방지하는 것이 더욱 중요시 된다.

3. 동작과정

우선 에이전트는 하부노드에 대하여 신규 추가되거나 제거된 노드에 대해 SNMP를 이용하여 확인을 하게 된다. 확인하여 수집된 노드정보는 관리자에게로 전송된다.

정책을 에이전트가 받아서 자신이 관리하는 하위 노드의 사양과 기종에 맞게 변환시키는 과정이 필요하며 이

때 에이전트는 각 하위 노드의 특징에 맞도록 처리를 할 수 있는 기능을 에 맞게 재작되어야 하고 각각의 노드는 논리적으로 하나의 에이전트와 연결되어야 하지만 하나의 에이전트가 여러 하부노드를 모두 관리할 수도 있다. 이렇게 연결된 에이전트들은 정책을 각 노드의 특성에 맞게 변형시켜 자신의 도메인에 적용되어야 할 정책들을 전달하고 수용되도록 한다. <그림 4>은 정책에 따른 에이전트의 동작 과정을 나타낸 것이다.



<그림 4> 정책에 따른 에이전트의 동작 과정

정책들을 각 노드에 적용시키기 위해 에이전트는 하부 노드와 적절한 협상을 거쳐 실제 노드의 Rule-Set에 적용시킬 준비를 한다. 에이전트가 원하는 경우 언제든지 노드와 협상 과정을 거쳐 노드의 Rule-Set을 새로운 정책에 따른 Rule-Set으로 바꾸고 적용하여 새로운 동작을 시키게 된다. 각각의 노드는 거의 항상 동작 중일 가능성이 높으므로 에이전트와 노드는 주기적인 통신을 통하여 서로간의 존재와 위치 상태 등을 항시 감시하게 되고 일련의 동작을 통해 얻은 정보를 기준으로 하여 상위 혹은 다른 도메인에서 전달한 정책들을 각 노드의 도메인에 적용시켜 동작을 하게 된다. 예를 들어 어느 한 도메인의 호스트에서 바이러스가 네트워크로 전파되려고 할 때 기존에 정책들 설정되어 수행되고 있다면 이 바이러스는 다른 도메인 혹은 상위의 도메인에게 전달되지 못할 것이고 이때 자신의 도메인에 속한 호스트에게만 피해를 주게 된다. 또한, 정책이 적용되기 전에 해당 에이전트의 하부에서 문제 발생시 동일 도메인내의 인접한 에이전트들 간에 P2P 형태로 이전에 수용한 정책을 기반으로 발생한 문제에 대해 관리자의 정책 설정에 앞서 문제점이 다른 하부 노드와 상위 노드에 전파되지 않도록 서로 간에 정보 교환과 정책 설정을 한다.

4. 결론

본 논문에서는 네트워크를 구성하는 라우터, 방화벽, 호스트, AP등의 다양한 노드관리를 위하여 에이전트의 정책 변환 기능과 에이전트 동적인 처리 과정등에 대한 특징을 분석하였으며, 네트워크에서 발생하는 노드관리의 문제점을 정의하고 이를 해결하기 위하여 P2P를 이용하여 에이전트간 협동처리 기능을 추가하고 SNMP를 이용하여 동적인 노드관리가 가능한 에이전트를 설계하였다.

이러한 에이전트를 통해 다양하고 산재되어 있는 네트워크 상에서의 노드에 대한 효율적 보안관리가 가능하며, 다양한 하위 노드로 구성되는 네트워크 적용에 대한 확장성 제공 및 P2P 기술을 적용함으로써 하부노드에서 발생하는 문제에 대하여 동일 도메인과 상위 네트워크로의 문제 전파를 미연에 방지하기 위하여 인접 노드들 간에 스스로 처리를 할 수 있는 환경을 제공한다.

또한, 향후 다양한 네트워크 환경에 적용할 수 있는 정책에 대한 확장과 문제발생시 이를 미연에 방지하기 위한 에이전트들 간에 P2P를 이용한 협동 처리 범위를 확대하고 보다 많은 시스템 정보를 수집하여 정책을 적용함에 있어서의 문제점 해결에 대한 연구를 진행할 예정이다.

[참고 문헌]

[1] Ribeiro,C., A Zuquete, "SPL: An access control language for security policies with complex constraints," NDSS, 2001.

[2] A. V. Aho, R. Sethi, J. D. Ullman , "Compilers: Principles, Techniques, & Tools," Addison Wesley, 1986

[3] K. C. Louden, "Compiler Comstruction Principles and Practice," PWS Publishing Company, 1997

[4] J.Levine, "Lex & Yacc," O'Reilly, 2001

[5] W. R. Stevens, "Unix Network Programming", Prentice Hall, 1998

[6] M. Subramanian, "Network Management : Principles and Practice", Addison Wesley, 2000

[7]Andy Oram, "Peer-to-Peer : Harnessing the Power of Disruptive Technologies", O'Reilly, 2001

[8] David Barkai, "Peer-to-Peer Computing: Technologies for Sharing and Collaborating on the Net", Intel Press, 2002

[9] Gheorghe Tecuci, "Building Intelligent Agents : An Apprenticeship, Multistrategy Learning Theory, Methodology, Tool and Case Studies", Academic Press, 1998