情報保護學會論文誌
第12券 第2號, 2002. 4

암호 기술을 이용한 안전한 능동 패킷 전송

김 영 수*, 나 중 찬*, 손 승 원*

A Secure Active Packet Transfer using Cryptographic Techniques

Youngsoo Kim*, Jungchan Na*, Seungwon Sohn*

요 약

능동 네트워크는 네트워크 구조에 대한 새로운 접근 방식이다. 노드(라우터 또는 스위치)들이 사용자 데이터에 대한 계산을 수행할 수 있는 반면, 패킷이 노드 상에서 실행될 프로그램을 운반하고 잠재적으로 해당 노드들의 상태를 변화시킬 수도 있다. 능동 네트워크는 매우 강력한 융통성을 갖는 반면, 기존의 네트워크보다 복잡할 뿐 아니라 보안상의 많은 문제점을 가지고 있다. 노드들은 공개된 자원이고 많은 중요 시스템들을 정확하게 실행시켜야 하므로, 패킷의 코드가 실행될 수 있는 계산 환경에서의 보안 요구 사항들은 매우 엄격하게 정의되어야 한다. 능동 네트워크의 보안 관련 연구는 크게 능동 노드 보호와 능동 패킷 보호로 나누어 볼 수 있다. 패킷 인증(authentication)이나 모니터링/제어(monitoring/control) 등이 전자에 속하는 반면, 암호화 기술(cryptographic techniques)을 이용하는 방법이 후자에 속한다. 본 논문은 암호화 기술을 이용한 안전한 능동 패킷 전송에 관한 것으로, 능동 패킷들을 이웃한 능동 노드들에게 안전하게 전송하고 해당 패킷들에 포함된 실행 코드들을 각 능동 노드 상에서 실행시킬 수 있는 새로운 방법을 제안한다. 여기서 제안하는 시스템은 공개키 암호 방식과 관용 암호 방식을 함께 사용한다.

ABSTRACT

Active networks represent a new approach to network architecture. Nodes(routers, switches, etc.) can perform computations on user data, while packets can carry programs to be executed on nodes and potentially change the state of them. While active networks provide a flexible network infrastructure, they are more complex than traditional networks and raise considerable security problems. Nodes are public resources and are essential to the proper and correct running of many important systems. Therefore, security requirements placed upon the computational environment where the code of packets will be executed must be very strict. Trends of research for active network security are divided into two categories: securing active nodes and securing active packets. For example, packet authentication or monitoring/control methods are for securing active node, but some cryptographic techniques are for the latter. This paper is for transferring active packets securely between active nodes. We propose a new method that can transfer active packets to neighboring active nodes securely, and execute executable code included in those packets in each active node. We use both public key cryptosystem and symmetric key cryptosystem in our scheme

keyword : *능동 네트워크, 능동 노드, 능동 패킷, 암호화 기술*

Ⅰ. 서 론

기존 네트워크의 기능은 패킷들을 하나의 종단점에서 다른 종단점으로 전달하는 것이었다. 이러한 시스템에서는 네트워크가 행하는 작업과 사용자가 행하는 작업에 명백한 구분이 있었다. 네트워크가 수행하는

* 한국전자통신연구원 네트워크보안연구부(blitzkrieg@etri.re.kr)

작업은 기본적으로 라우팅, 혼잡 제어(congestion control), QoS 서비스 스킴등으로 제한되었다. 이러한 네트워크는 '수동적(passive)'이라 할 수 있다. 수동적 네트워크는 많은 문제점들을 갖고 있다. 여러 프로토콜 계층에서의 너무 많은 동작으로 인하여 성능이 저하되고, 분리된 네트워크 환경에서 각기 존재하는 표준들과 새로운 기술들의 통합이 어려우며, 기존의 구조위에 새로운 서비스를 추가하는 것 또한 매우 어렵다. 방화벽, 웹 프록시, 멀티캐스트 라우터 및 이동형 프록시 등 네트워크 내에서도 계산을 요하는 응용들의 출현 또한 기존의 수동적 네트워크를 더욱더 어렵게 만들고 있다. 네트워크 구조 자체의 지원 없이, 이들 응용들은 네트워크 상의 노드에서 사용자-조작 계산을 수행하는 경험적(ad-hoc) 서비스들을 적용시켜왔다. 수 많은 경험적 접근 방식을 사용자들이 자신의 네트워크를 프로그래밍 할 수 있도록 하는 네트워크-기반 계산으로 대신할 필요성을 느끼게 되었고, 사용자에게 네트워크를 프로그래밍할 수 있는 능력을 부여하는 혁신적 아이디어를 능동 네트워킹(active networking)[1]이라 부른다.

능동 네트워크는 네트워크 구조에 대한 새로운 접근 방식이다. 이러한 네트워크는 두가지 차원에서 '능동적'이다: 네트워크 내의 라우터나 스위치들은 그들 사이를 흐르는 사용자 데이터에 대한 계산을 수행할 수 있다; 그리고 사용자들은 이러한 계산들을 위해 그들 자신의 프로그램들을 제공함에 의해 네트워크를 프로그래밍할 수 있다.

한편, 능동 네트워크는 기존의 네트워크에 비하여 융통성(flexibility)이 매우 큰 반면, 많은 보안[1](security) 문제들을 안고 있다[2]. 실행 가능 코드를 담고 있는 능동 패킷은 노드의 상태를 변화시킬 수 있다. 노드들은 공개된 자원이고 많은 중요 시스템들을 정확하게 실행시켜야 하므로, 패킷의 코드가 실행될 수 있는 계산 환경에서의 보안 요구 사항들은 매우 엄격하게 정의되어야 한다.

능동 네트워크의 보안 관련 연구는 크게 능동 노드 보호와 능동 패킷 보호로 나누어 볼 수 있다. 능동 패킷 인증(authentication)이나 모니터링/제어(monitoring/control) 등이 전자에 속하는 반면, 암호화 기술(cryptographic techniques)을 이용하는 방법이 후자에 속한다.

본 논문은 암호화 기술을 이용한 안전한 능동 패킷 전송에 관한 것으로, 송신 노드가 수신지를 알고 있다는 가정 하에 행하는 기존의 암호 프로토콜들과는 달리, 능동 네트워크의 특성상 가변적인 수신 노드와의 안전한 패킷 전송을 위하여 기존 프로토콜을 개선하였다. 제안하는 시스템을 통하여 송신 노드는 능동 패킷들을 이웃한 능동 노드들(종단 노드와 중간 노드, 중간 노드들)에게 안전하게 전송하고 해당 패킷들에 포함된 실행 코드들을 각 능동 노드 상에서 실행시킬 수 있다. 제 2장에서는 우선 능동 노드 및 능동 패킷이 범할 수 있는 오용의 문제점들과 기존의 해결책을 능동 노드 보호 관점과 능동 패킷 보호 관점으로 나누어 간략히 살펴본다. 제 3장부터 제 7장까지는 능동 패킷 보호 관점에서의 해결책이며 본 논문의 주제인 암호 기술을 이용한 안전한 능동 패킷 전송에 관한 부분이다. 제 3장에서는 우리가 제안하는 시스템을 위한 능동 네트워크 환경을 설정하고, 능동 패킷의 포맷을 기술하였다. 제안하는 시스템에 대한 이해를 돕기 위한 기본 가정들과 표기를 제 4장에 나타내었고, 제 5장에서는 시스템을 제안하고 그 동작 과정들을 상세하게 설명하였다. 제 6장은 제안하는 시스템과 기존 프로토콜을 비교하고, 제 7장에서 결론을 맺는다.

II. 능동 패킷 및 노드 오용의 문제점과 기존의 해결책

본 장에서는 우선 능동 패킷이나 능동 노드가 범할 수 있는 오용(misuse)의 문제점들을 알아보고, 능동 네트워크가 가진 보안 문제을 위한 기존 방법들을 능동 노드 보호 관점과 능동 패킷 보호 관점으로 나누어 간략하게 살펴본다.

2.1 능동 패킷 및 능동 노드가 범할 수 있는 오용의 문제점

능동 네트워크에서, 능동 패킷들은 능동 노드, 네트워크 자원 및 다른 방법으로 동작하는 능동 패킷들을 오용할 수 있다. 또한, 능동 노드들도 능동 패킷을 오용할 수 있다. 이동 소프트웨어 에이전트의 보안 이슈들과 관련한 이전의 연구들이 여기서도 적용될 수 있는데, 다음은 이와 관련하여 발생 가능한

1) 안전성(safety)이 실수나 의도하지 않는 동작에 대한 리스크를 줄이는 것인 반면, 보안(security)은 비밀성(privacy) 및 무결성(integrity) 보장과 악의적 공격으로부터의 보호를 의미한다.

문제들이다[3]:

- 손상(damage): 능동 패킷은 메모리로부터 자원이나 노드의 서비스를 재구성, 변경 또는 제거함에 의해 자원이나 노드의 서비스들을 파괴 또는 변경할 수 있다. 노드는 능동 패킷이 자신의 노드에서 작업을 마치기 전에 지울 수 있다. 또한, 동일한 계산 환경을 갖는 능동 패킷들은 서로를 공격할 수 있다.
- 서비스 거부(denial of service): 능동 패킷은 지속적으로 네트워크 연결을 유지하거나 가능한 CPU 시간의 많은 부분을 사용함에 의해 자원이나 서비스를 오버 로딩할 수 있다. 노드는 이러한 상황에서 정확한 기능을 할 수 없으며 다른 능동 패킷들은 실행되거나 포워딩되지 못한다.
- 서비스 도용(theft of service): 능동 패킷은 노드에서 비밀 정보에 접근하여 이를 가져갈 수 있는 반면, 노드 방문 시 여러 곳에서 노드에 대해 취약성이 노출되어 있다.
- 혼합 공격(compound attack): 실제적으로 능동 노드의 가장 심각한 위협은 혼합 공격이다. 예를 들어, 악의적 사용자는 많은 능동 패킷들을 중앙 라우터에게 보내어 대역폭 용량 소비로 인하여 다운되도록 할 수 있다.

2.2 능동 노드 보호

능동 노드를 보호하기 위해 다음과 같은 방법들이 제안되었다.

- 능동 패킷의 인증(authentication): 능동 패킷은 공개키 기반의 전자 서명(digital signature) 알고리즘 등을 사용하여 생성한 확인서(credential)를 가져야 한다. 이러한 확인서는 유용하지만 단순히 능동 패킷에 대한 신원을 보장해 줄 뿐 능동 패킷이 유해하지 않다는 것을 보장하지는 못한다.
- 모니터링(monitoring)과 제어(control): 능동 패킷의 사용과 접근이 허용된 정보, 시스템 자원 및 서비스를 제한하기 위하여 참조 모니터(reference monitor)를 사용한다. 참조 모니터는 접근이 허락되었는지를 결정하기 위하여 보안 정책(security policy)을 참조한다. 접근 단계 모니터링은 패킷이 무엇을 할 수 있는지를 직접 제한할 수 있으므로 유용하게 사용될 수 있다. 그러나 어떤 자원의 사용 허락을 받기 위한 결정이 이미 언급한 바와 같이 패킷 자체가 유해하지 않다는 것을 보장할 수 없는 인증서를 기반으로 하기 때문에 이 방법 또한 완벽하지는 않다.
- 제한 기법(limitation techniques): 능동 패킷의 허용 실행 시간을 제한하거나, 패킷이 통과할 수 있는 전체 노드의 수를 한정하거나, 또는 패킷이 중복될 수 있는 횟수를 제한하는 등 노드의 자원을 특정 능동 패킷이 독점하여 사용하지 못하도록 하는 방법이다.
- 검증 전달 코드(Proof Carrying Code, PCC): PCC[4]는 결과를 도출해 내는 것보다 그것이 맞는지를 확인하는 것이 더 쉽다는 점에서 착안했다. 이것은 정확성을 검증할 수 있는 부분을 능동 패킷에 삽입하여 전송하면 능동 노드는 그것을 쉽게 검증한 후 프로그램을 실행한다. 어려운 부분은 검증을 생성하는 것인데, 이것은 프로그래머가 담당하여야 할 부분이다.

2.3 능동 패킷 보호

능동 패킷의 보호를 위해 암호 기술(cryptographic techniques)과 결함 내구성(fault-tolerance) 기법 등 두 가지 방법이 제시되었다.

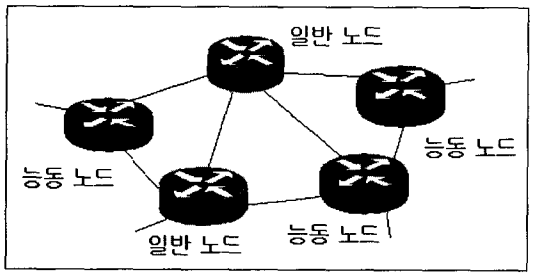
- 암호 기술: 기존 네트워크에서의 암호 기술은 전체 네트워크 경로를 모두 보호해야 할 지라도 두 종단(end node)간 즉, 송신지와 수신지간에서만 암호화 과정이 수행되었으며, 미리 정의된 위험을 기반으로 보호하는 방법을 사용하였다. 그러나 능동 네트워크 패러다임에서 능동 패킷에는 일반적으로 프로그램 코드와 데이터가 전송되므로 기존의 종단간 암호화 기법을 적용하기에 적합하지가 않다[5]. 기존의 암호화 기법은 종단간에서만 암호화하는 반면, 능동 네트워크에서는 능동 패킷의 특성으로 인해 내용에 접근할 필요가 있는 중간 노드(intermediate node)들에 대해서도 암호화를 해야 하고, 패킷이 경유하는 경로가 고정되어 있지 않고 가변적이므로 미리 정의된 위험을 기반으로 보호할 수 없다.
- 결함 내구성: 결함 내구성 기법에는 복사(replication), 보존(persistence), 리디렉션(redirection)이 있다. 복사란 노드에서 오류가 발생할 경우를

대비하여 각 노드에 패킷들을 복사하는 것이고, 보존은 노드가 파괴되는 등의 노드 결함에 대비하여 패킷들을 임시로 저장하여 복사본을 저장 공간에 유지하는 것이다. 리디렉션은 디폴트 경로(default route)가 실패했을 경우 다른 경로를 찾는 것이다.

복사 및 보존은 메모리와 대역폭을 많이 소모하므로 대부분의 네트워크 패킷 보호에는 부적합한 방법이다. 리디렉션과 암호화 기법은 기본적으로 CPU 자원만을 소모하므로 패킷 보호에 가장 널리 쓰인다. 따라서 능동 패킷을 보호하는 문제는 결함 내구성 기법과 암호 기술을 조합하여 사용하는 것이 좋다. 그러나 이러한 기법들이 아직 초기 단계에 있으므로 확실하고 완벽하게 보호하려면 더 많은 연구가 필요하다.

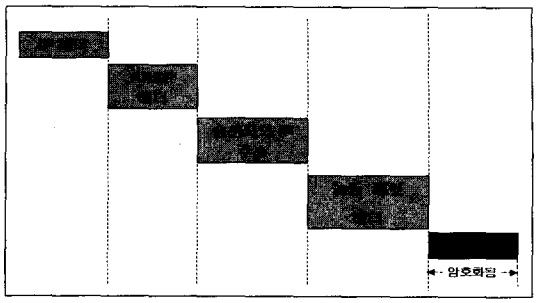
Ⅲ. 네트워크 설정 및 능동 패킷 포맷

[그림 1]은 제안하는 시스템을 위한 네트워크 구성을 나타낸다. 본 네트워크 환경에서는 능동 노드와 기존의 일반 노드가 함께 존재한다. 만일 일반 노드가 능동 패킷을 받으면 이를 이웃 노드로 단순히 포워딩한다.

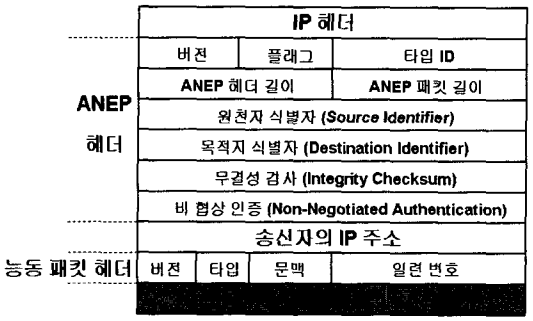


[그림 1] 네트워크 설정

하나의 능동 노드(송신 노드)가 다음 능동 노드에 안전하게 전달해야할 실행 가능 코드를 가지고 있을 경우, 송신 노드는 능동 패킷을 생성하고, 패킷의 페이로드 부분에 암호화한 부분을 담아 이웃 능동 노드가 수신할 수 있도록 이를 포워딩한다. [그림 2]는 대략적인 능동 패킷 포맷을 나타내었고, [그림 3]은 자세한 능동 패킷 포맷을 스마트 패킷[6]을 예로 들어 나타낸 것이다.[2]



[그림 2] 대략적인 능동 패킷 포맷



[그림 3] 상세한 능동 패킷 포맷

3.1 ANEP 헤더

ANEP(Active Network Encapsulation Protocol)[7]은 타 매체로의 전송을 위한 능동 네트워크 프레임 캡슐화에 관한 메커니즘을 명시한 것으로, 본 프로토콜의 포맷은 기존의 네트워크 기반 구조(IP 또는 IPv6)나 링크 계층에서의 전송에 사용될 수 있다. ANEP 헤더는, 패킷을 수신한 능동 노드가 자신이 수행될 환경을 혼동없이 신속하게 결정할 수 있도록 하고, 캡슐화된 프로그램에 담기에 적합하지 않은 정보들을 헤더에 위치시켜야 하며, 의도한 수행 환경(Execution Environment)을 사용하지 못하게 될 경우, 패킷들이 디폴트로 수행할 수 있도록 하기 위해 필요하다. ANEP 헤더는 버전 필드(version field), 플래그 필드(flags field), ANEP 헤더 길이 필드(ANEP header length field), 타입 아이디 필드(type ID field), ANEP 패킷 길이 필드(ANEP packet length field) 및 옵션 필드들로 이루어져 있다. 버전 필드는 사용중인 헤더 포맷을 나타내는 필드로, ANEP 헤더가 수정될 경우 변경되는데, 하나의 능동 노드가 인식할 수 없

2) [그림 3]에서 송신자의 IP 주소 부분은 ANEP 헤더의 원천지 식별자 필드로 대체될 수 있다.

는 버전 번호를 가진 패킷을 수신할 경우 그 패킷은 버린다. 플래그 필드의 값이 0이면, (필요한 정보가 헤더의 옵션 부분에 들어 있는 경우) 노드는 디폴트 라우팅 메커니즘을 사용하여 패킷을 포워딩하고, 값이 1이면 패킷을 버린다. ANEP 헤더 길이 필드는 ANEP 헤더 길이를 32 비트 워드 단위로 명시한다. 타입 아이디 필드는 메시지의 수행 환경을 나타내는 필드로, 해당 수행 환경에 타입 아이디 값을 부여하는 기관은 ANANA(Active Networks Assigned Numbers Authority)이다. 타입 아이디 값 0은 향후 네트워크 계층 정보와 에러 메시지를 위해 예약되어 있다. ANEP 패킷 길이 필드는 패킷 페이로드를 포함한 전체 패킷 길이를 옥텟 단위로 나타낸 것으로, 패킷 길이를 재구성할 수 없는 하위 계층에 전송될 경우 패킷을 재구성하는데 사용된다.

원천지 식별자(source identifier), 목적지 식별자(destination identifier), 무결성 검사(integrity checksum) 및 비-협정 인증(non-negotiated authentication) 등 4개의 필드는 옵션 부분[3]이다.

원천지 식별자 옵션은 능동 네트워크 상에서 패킷 송신자를 고유하게 식별하는 값을 포함하는 것으로, 이 옵션의 페이로드는 사용되고 있는 주소 지정 스킴과 해당 스킴의 데이터를 나타내는 하나의 32 비트 값으로 구성된다. IPv4 주소가 1, IPv6 주소가 2, 그리고 802.3 주소가 3으로 예약되어 있다. 목적지 식별자 옵션은 패킷의 최종 목적지를 유일하게 식별하는 값을 포함하는 것으로, 페이로드 포맷은 원천지 식별자의 그것과 동일하다. 이 필드는, 의도한 수행 환경을 사용할 수 없을 경우, 패킷을 좀 더 잘 처리할 수 있는 능동 노드에게 포워딩하기 위해 사용될 수 있다. 무결성 검사 옵션의 페이로드는 체섬(checksum) 계산을 위해 0으로 셋팅되어야 한다. 비-협정 인증 옵션은, 패킷 생성자와 프로세싱 노드(들) 간의 우선적 협상(prior negotiation) 없이, 일방향 인증을 제공하는데 사용된다. 이 옵션의 페이로드는 사용중인 인증 스킴과 그 내용을 나타내는 하나의 32비트 값으로 구성되는데, 인증을 요청하는 패킷의 개수가 전체 협상의 비용에 비하여 너무 작을 때, 동작 시간이 중요시될 때, 또는 안전성 협상이 발생할 수 없을 때 사용될 수 있다.

3) 최근에 공개된 능동 네트워크 보안 워킹 그룹(Active network security working group)의 문서[8]에서는 본문에 열거한 4가지 예약된 값들 외에 홉-무결성(Hop Integrity)이나 인라인 정책(In-line policy) 등 다른 옵션들도 다수 정의하고 있다.

3.2 능동 패킷 헤더

능동 패킷 헤더는 ANEP의 페이로드에 포함되는 부분 중 하나이며, 각 수행 환경에 따라 그 값들의 정의가 다를 수 있다. 위에서도 언급하였듯이 [그림 3]은 스마트 패킷에서 정의한 이른바 스마트 패킷 헤더(smart packet header)의 필드들을 하나의 예로 나타낸 것이다. 버전 필드(version field)는 언어 업그레이드와 패킷 포맷 변경을 나타내고, 타입 필드(type field)는 패킷의 타입을 구분한다. 패킷은 프로그램 패킷(program packet), 데이터 패킷(data packet), 에러 패킷(error packet) 및 메시지 패킷(message packet) 등 4가지 타입을 갖는다. 프로그램 패킷은 특정 호스트에서 실행되는 코드를 담고 있고, 데이터 패킷은 실행 결과를 원래의 네트워크 관리 프로그램으로 반환하는 역할을 하며, 에러 패킷은 프로그램 패킷의 전송이나 코드 실행 중 예외 상황이 발생할 경우 에러 상태를 반환한다. 끝으로, 메시지 패킷은 정보를 담은 메시지를 운반한다. 문맥 필드(context field)는 스마트 패킷의 생성자를 나타내는 값을 담고 있는 필드로, 각 클라이언트의 ANEP 데몬에 의해 생성되며, 그 호스트의 특정 클라이언트에 대해 고유하다. 이 값은 프로그램 패킷에 위치하는데, 프로그램 패킷이 네트워크를 통과하면서, 하나 또는 여러 개의 응답(데이터, 에러 또는 메시지 패킷)을 생성하고, 문맥 값은 응답이 전달되어야 하는 클라이언트를 구분하는데 사용된다. 일련 번호 필드(sequence number field)는 동일 문맥의 메시지들을 구분하는데 사용되는 값을 담고 있다.

Ⅳ. 표기와 가정

본 장에서는 제안하는 시스템의 동작 흐름을 잘 나타낼 수 있도록 몇 가지 객체들에 대한 간략화된 표기(notation)를 열거하고, 필요한 기본 가정들을 기술한다.

4.1 표기

혼동을 막기 위해 공개키 암호 방식과 대칭키 암호 방식의 키 및 동작을 구분하여 표기하였다.

- KE_A : 능동 노드 A의 공개키(public key) (RSA[9] 등과 같은 공개키 암호 방식에서의 암호화키)
- KD_A : 능동 노드 A의 개인키(private key) (공개키 암호 방식에서의 복호화키)
- KS_A : 이웃 능동 노드와의 안전한 통신을 위해 필요한 능동 노드 A의 비밀 세션키(session key) (Triple-DES[10]와 같은 대칭키 암호 방식에서의 암/복호화키). 안전성을 높이기 위해서는 반복 사용 없이 일회 사용 후 변경하는 것이 바람직하다.
- CERT_A : 능동 노드 A의 인증서(certificate) (공개키 암호 방식)
- PGM : 능동 패킷에 담긴 프로그램(코드)으로 암(복)호화의 대상
- P : 패킷의 페이로드 부분
- MD(P) : P에 대한 메시지 다이제스트 값 (MD5[11]와 같은 해쉬 함수 사용)
- $ENC_X(Y)$: 공개키 암호 방식을 이용하여 Y를 키 X로 암호화(encryption)
- $DEC_X(Y)$: 공개키 암호 방식을 이용하여 Y를 키 X로 복호화(decryption)
- $E_X(Y)$: 대칭키 암호 방식을 이용하여 Y를 X로 암호화
- $D_X(Y)$: 대칭키 암호 방식을 이용하여 Y를 X로 복호화
- $Sig_X(Y)$: 디지털 서명 스킴(Schnorr 서명[12], RSA 서명 등)을 이용하여 Y를 키 X로 서명
- VER(S) : 서명 S에 대한 검증
- CA : 인증 기관(certification authority)
- K_CA : 인증서 발급을 위한 CA의 키
- INFO: 인증서에 포함된 정보들. 예를 들면, 사용된 암호 알고리즘, 유효 기간, 발급 기간 등
- REQ(Y): Y를 얻기 위한 요청 메시지

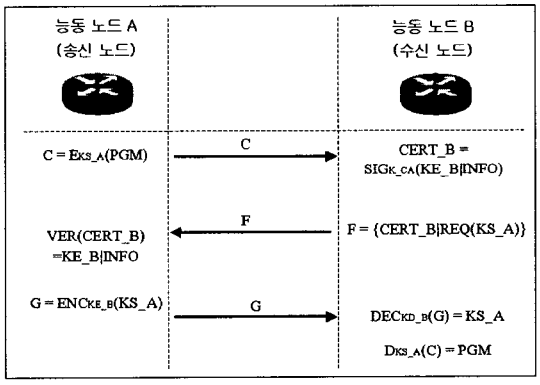
4.2 가정

제안하는 스킴이 성립하기 위해서는 다음과 같은 가정이 필요하다.

- 능동 노드들과 일반 노드들이 함께 존재하며, 이웃하는 노드들의 주소는 서로 알지 못한다.
- 기존 노드들은 능동 패킷을 받으면, 이를 무시하고 단순히 포워딩만을 한다.
- 각 능동 노드는 비밀 세션키(대칭키 암호 방식에 사용)와 공개키/개인키쌍(공개키 암호 방식에 사용)을 갖는다.
- 각 도메인 마다 하나의 인증 기관(Certificate Authority, CA[13])이 존재하고, 각 노드들은 해당 인증 기관에 등록되어 있다고 가정한다. (여기서 도메인은 네트워크나 서브네트워크를 의미한다.)
- 인증 기관은 신뢰할 수 있는 제삼자(Trusted-Third-Party, TTP[14])라고 가정한다.
- 등록시 인증기관은 각 능동 노드들에게 자신의 개인키로 디지털 서명한 인증서(certificate)를 제공한다. (단, 여기서 사용하는 디지털 서명은 Schnorr 서명이나 RSA 서명 같은 위조 불가한 (unforgeable) 서명 스킴을 사용한다고 가정한다.)
- 종단(end-to-end) 능동 노드사이에 존재하는 모든 중간(intermediate) 능동 노드 상에서 해당 능동 패킷 내의 프로그램이 실행되어져야 한다고 가정한다.

V. 제안하는 스킴

[그림 4]는 능동 노드 A와 능동 노드 B 사이에 일어나는 동작들의 흐름을 나타낸 그림이다. 능동 노드 A는 이전 능동 노드로부터 받았거나 또는 그 자신이 생성한 코드(PGM)를 최종 목적지 노드에 보내고, 중간에 만나게 되는 능동 노드들이 모두 이 코드를 실행할 수 있도록 하고자 한다. 세부 동작 과정은 다음과 같다.

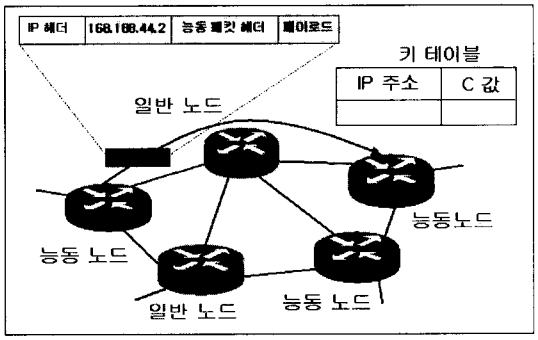


[그림 4] 동작 흐름

(1) 능동 노드 A(송신 노드)는 자신의 비밀 세션키 KS_A를 생성한다. 안전하게 전송하고자 하는 프로그램(PGM)이 담긴 능동 패킷을 대칭키 암호 방식을 이용하여 암호화하고 그 값을 담은 능동 패킷을 생성한다.

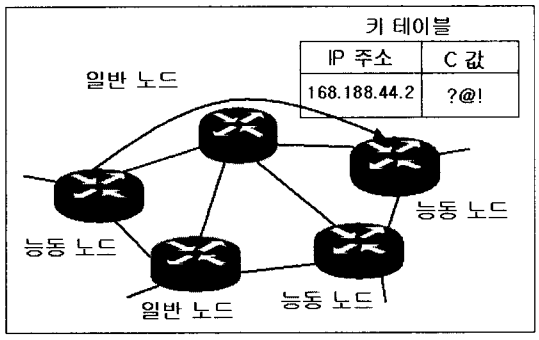
$$C = E_{KS\_A}(PGM) \quad (1)$$

(2) 능동 노드 A는 목적지 주소 외에 다른 노드들의 주소를 알지 못하므로 C를 이웃 노드로 포워딩 한다. ([그림 5] 참조)



[그림 5] 능동 패킷 도착

(3) 암호화된 능동 패킷 C를 수신한 능동 노드 B (수신 노드)는 헤더를 통해 C가 능동 패킷임을 검사하고 자신의 키 테이블에 송신자의 IP 주소와 키 값을 기록한다 ([그림 6] 참조)



[그림 6] 키 테이블에 값들을 기록

(4) 암호화된 값이 들어있는 C를 복원하기 위해서는 비밀 세션키 KS_A가 필요하므로 능동 노드 B는 송신자 IP 주소로 보낼 정보를 생성한다. 우선 자신의 공개키와 사용된 알고리즘 등을 담은 정보 INFO 등이 인증 기관 CA의 개인키 K_CA로 디지털 서명된 자신의 인증서 CERT_B를 준비한다.

$$CERT\_B = SIG_{K\_CA}(KE\_B|INFO) \quad (2)$$

(5) 능동 노드 B는 CERT_B와 KS_A값을 요청하는 요청 메시지 REQ(KS_A) 등을 담은 패킷 F를 위에서 기록해 둔 송신자 IP 주소로 전송한다.

$$F = \{CERT\_B \mid REQ(KS\_A)\} \quad (3)$$

(6) F를 수신한 능동 노드 A는 헤더를 통해 이를 체크한 후, 능동 노드 B의 인증서 CERT_B를 CA의 공개키로 검증하고 B의 공개키 KE_B를 추출한다.

$$VER(CERT\_B) = KE\_B|INFO \quad (4)$$

(7) 자신의 키 테이블에 송신자 IP 주소와 공개키 값 KE_B, 인증서 값 CERT_B 등을 기록한다.

(8) 능동 노드 A는 추출한 능동 노드 B의 공개키 KE_B로 자신의 비밀 세션키 KS_A를 공개키 방식으로 암호화한 값 G를 능동 노드 B에게 전송한다.

$$G = ENC_{KE\_B}(KS\_A) \quad (5)$$

(9) 이를 수신한 능동 노드 B는 자신의 개인키 KD_B를 이용하여 G를 풀고, 이로 인해 획득한 KS_A로 처음에 받은 능동 패킷 C를 복호화하여 목적한 프로그램 PGM을 복원한다.

$$DEC_{KD\_B}(G) = KS\_A \quad (6)$$

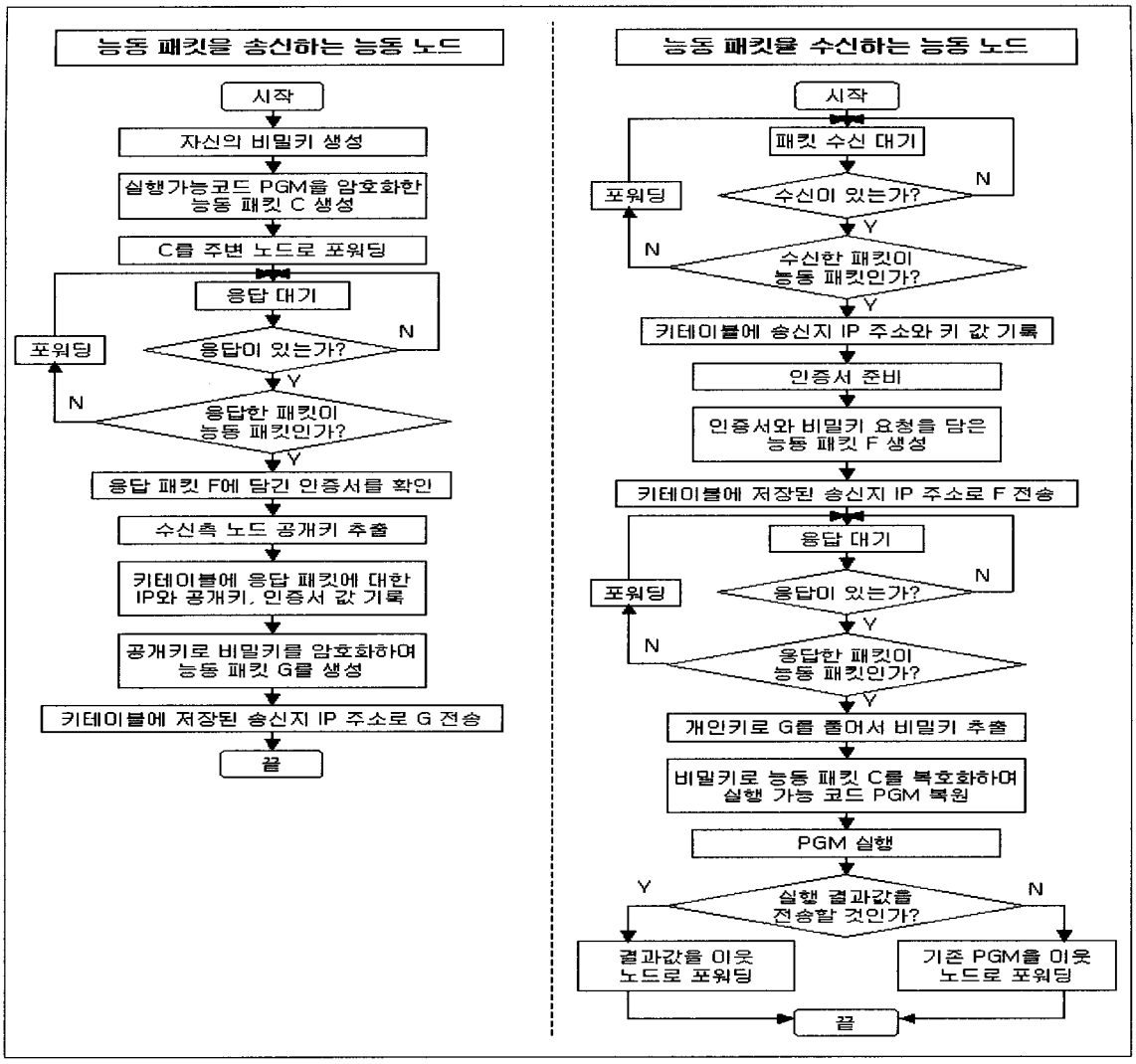
$$D_{KS\_A}(C) = PGM \quad (7)$$

(10) 능동 노드 B는 프로그램을 실행하고 그 결과값 또는 능동 노드 A로부터 수신한 프로그램을 다시 이웃 능동 노드로 안전하게 전송하기 위해 위의 과정을 반복한다.

[그림 7]은 위의 동작 과정을 송신측과 수신측으로 분리하여 나타낸 그림이다.

Ⅵ. 기존 해결책과의 비교

본 장에서는 기존 네트워크 환경에서 적용 가능하고 제안하는 시스템과 유사하나, 능동 네트워크 환경에서는 적용할 수 없는 암호 기술인 전자 봉투(digital envelope) 프로토콜[15]을 간략하게 살펴보고, 제안하는 시스템과 비교해 보고자 한다.



〔그림 7〕 송신/수신 노드에서의 동작 흐름

6.1 전자 봉투

전자 봉투 프로토콜에서 송신 노드와 수신 노드는 다음과 같은 과정을 통해 패킷을 안전하게 전달할 수 있다. (단, 상대방의 공개키가 필요하므로, 송신 노드가 수신 노드를 미리 알고 있고, 공개키 디렉토리로부터 수신 노드의 공개키를 수신한 상태라고 가정한다.)

(1) 송신 노드 A는 일반 패킷의 페이로드 부분을 지정된 수신 노드에게 안전하게 전송하고자 한다. (능동 패킷이 아니므로, 페이로드 부분은 실행 가능 코드가 아닌 데이터가 포함되어 있다) 송신 노드 A는 우선 페이로드 부분을 해쉬 함수를 이용하여 메시지 다이제스트 형태(=Q)로 만들고, 이를 자신의 개인키로 서명한 값 R을 계산한다.

$$MD(P)=Q \tag{8}$$

$$R=SIG_{KD\_A}(Q) \tag{9}$$

(2) 송신 노드 A는 곧이어 자신의 인증서(=CERT_A)와 P, 그리고 R을 묶어 비밀 세션키 KS_A로 암호화한다.

$$CERT\_A=SIG_{K\_CA}(KE\_A|INFO) \tag{10}$$

$$U = D_{KS\_A}(CERT\_A \mid P \mid R) \quad (11)$$

(3) 송신 노드 A는 비밀 세션키를 수신 노드 B의 공개키 KE_B로 암호화한 값 V를 계산한 후, U값과 V값을 수신 노드 B의 IP 주소로 전송한다.(여기서 V를 전자 봉투라 한다)

$$V = ENC_{KE\_B}(KS\_A) \quad (12)$$

(4) 이를 수신한 수신 노드 B는 우선 자신의 개인키 KD_B를 이용하여 전자 봉투를 복호화하여 송신 노드 A의 비밀 세션키 KS_A를 획득하고, 이 값으로 U를 복호화하여 A의 인증서, 페이로드 값 P, 그리고 서명값 R 등을 얻는다.

$$DEC_{KD\_B}(V) = KS\_A \quad (13)$$

$$D_{KS\_A}(U) = \{CERT\_A \mid P \mid R\} \quad (14)$$

(5) 수신 노드 B는 인증서에 담긴 송신 노드 A의 공개키 KE_A를 이용하여 R을 검증하고 페이로드 부분의 메시지 다이제스트 형태인 Q를 얻는다.

$$VER(R) = Q \quad (15)$$

(6) 끝으로, 수신 노드 B는 4 과정에서 얻은 P값을 해쉉하고 이를 통해 얻은 메시지 다이제스트 값과 5 과정에서 얻은 Q값이 일치하는지를 확인한다.

6.2 제안하는 시스템과의 비교

〔표 1〕은 제안하는 시스템과 전자 봉투 프로토콜을 비교한 것이다. 전자봉투 프로토콜의 경우, 송신 노드가 수신 노드의 공개키를 획득해야 하므로, 공개키 디렉토리에 해당 수신 노드의 공개키를 요청하고, 공개키를 수신하는 2회의 통신이 추가적으로 필요하다. 계산량의 경우, 전자 봉투 프로토콜이 제안하는 시스템에 비하여 1회의 서명 및 검증이 더 필요하다. (인증 기관이 인증서 생성시 행하는 서명 등의 계산은 두 방식 모두 동일한 조건이므로 무시하였다). 위에서도 언급하였듯이, 능동 네트워크에서는 능동 패킷의 특성으로 인해 내용에 접근할 필요가 있는 중간 노드들에 대해서도 암호화를 해야하고, 패킷이 경유하는 경로가 고정되어 있지 않고 동적으로 변경이 가능하므로 수신지를 미리 알아야 하는 기존의 전자 봉투 프로토콜 등은 적용할 수가 없다. 제안하는 시스템은 이러한 가변적 수신지 문제 해결을 위해 프로토콜 수행 중에 수신 노드가 자신의 공개키를 송신 노드에게 직접 전송해주는 방식을 택하였다.

7. 결 론

능동 노드들로 하여금 능동 패킷들을 안전하게 전송하고, 각 패킷에 담긴 실행 가능 코드들을 각 능동 노드 상에서 실행할 수 있는 새로운 스킴을 제안하였다. 우선 능동 네트워크의 개념과 그 유연성에 기인하는 몇가지 보안 관련 문제점들을 살펴보았다. 그리고, 기존의 암호 프로토콜을 변형한 새로운 시스템을 제안하고, 기존 프로토콜과 비교해 보았다. 늘어난 통신 횟수로 인하여 다소의 성능 저하가 예상되나, 능동 패킷이 경유하는 경로가 고정되어 있지 않고 동적으로 변경이 가능하므로, 통신 횟수의 증가는 불가피하다. 송신 노드가 수신 노드를 사전에 알고있다는 가정을 둘 수도 있겠으나, 이것은 능동 네트워크 패러다임에 어긋난 것이다. 본 논문에서는 성능에 대한 고려보다는 가변적 수신 노드에 적용할 수 있도록 기존 암호 프로토콜을 변형하는 것에 초점을 맞추었다.

〔표 1〕 제안하는 시스템과 전자 봉투 프로토콜간의 비교

		제안하는 시스템	전자 봉투 프로토콜	비고
통신횟수	송/수신노드간	3회	1회	
	송신노드와 공개키디렉토리간	-	2회	
계산량	(공개키방식)암/복호화	1회	1회	
	(대칭키방식)암/복호화	1회	1회	
	서명 및 검증	-	1회	
적용		수신지-가변적 환경	수신지-고정적 환경	

참 고 문 헌

[1] D.L. Tennenhouse, J.M. Smith, W.D. Sincoskie, D.J. Wetherall, and G.J. Minden, "A survey of active network research", IEEE Communications Magazine, Vol. 35, No. 1, pp. 80~86, 1997.

[2] K. Psounis, "Active networks: App. lications, Security, Safety, and Architectures", IEEE Communications Surveys, 1999.

[3] M.S. Greenberg, J.C. Byington, and D.G. Harper, "Mobile Agents and Security", IEEE Communications Magazine, Vol. 36, No. 7, 1998.

[4] G.C. Necula, "Proof-Carrying Code", Proc. Of 24th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, ACM Press, 1997.

[5] Y.S. Kim, J.C. Na, S.W. Sohn, "A Secure Method for Transferring Active Packets", Proc. of WSEAS'01, Cairns, Australia, Dec.17~21, pp. 259~262, 2001.

[6] B. Schwartz, A.W. Jackson,k W.T. Strayer, W. Zhou, R.D. Rockwell, and C. Partridge, "Smart Packets for Active Networks", BBn Technologies, 1998.

[7] D. Alexander, B. Braden, C.A. Gunter, A.W. Jackson, A.D. Keromytis, G.J. Minden, and D. Wetherall, "Active Network Encapsulation Protocol, Draft", Jan. 2000, available online at http://www.cis.upenn.edu/switchware/ANEP.

[8] DARPA AN Security Working Group, "Security Architecture for Active Nets", Nov.2001, available online at ftp://ftp.tislabs.com/pub/anfr/secrarch5.ps.

[9] R.L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public key cryptosystems", Communications of the Association for Computing Machinery, Vol. 22, No. 2, pp. 120~126, 1978.

[10] ANSI X9.17 (Revised), "American National Standard for financial Institution Key Management(Wholesale)", American Bankers Association, 1985.

[11] R.L. Rivest, "The MD5 Message Digest Algorithm", RFC 1321, 1992.

[12] C.P. Schnorr, "Efficient signature generation for smart cards", Proc.of Crypto '89, Springer-verlag, LNCS Vol. 435, pp. 239~252, 1990.

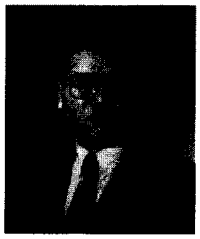
[13] B. Schneier, "App. lied Cryptography: Second Edition", Wiley, pp. 185~187, 1996.

[14] A.J. Menezes, P.C. Oorschot, and S.A. Vanstone, "Handbook of App. lied Cryptography", CRC Press, pp. 547~550, 1997.

[15] L. Loeb, "Secure Electronic Transactions: Introduction and Technical Reference", Artech House Publishers, 1998.

〈著者紹介〉

김 영 수(Youngsoo Kim) 정회원
1998년 2월 : 성균관대학교 정보공학과 졸업
2000년 2월 : 성균관대학교 컴퓨터공학과 석사
2000년 2월~현재 : 한국전자통신연구원 정보보호연구본부 연구원
〈관심분야〉 암호이론, 네트워크 보안



나 중 찬(Jungchan Na) 정회원
1986년 2월 : 충남대학교 계산통계학과 졸업
1989년 2월 : 숭실대학교 전산학과 석사
1989년 2월~현재 : 한국전자통신연구원 정보보호연구본부, 능동보안기술연구팀 팀장
/선임연구원
〈관심분야〉 실시간 시스템, 분산시스템, 네트워크, 정보보호

손 승 원(Seungwon Sohn) 정회원
1984년 2월 : 경북대학교 전자공학과(공학사)
1994년 2월 : 연세대학교 산업 대학원 전자공학과(공학석사)
1999년 2월 : 충북대학교 대학원 전자공학과(공학박사)
1991년 8월~현재 : 한국전자통신연구원 정보보호연구본부, 네트워크보안연구부 부장
/책임연구원
〈관심분야〉 네트워크 보안, 라우팅 알고리즘