情報保護學會論文誌
第13券 第2號, 2003. 4

이중 해쉬체인에 기반을 둔 Link-State 라우팅 보안 메커니즘

유 병 익* , 임 정 미* , 유 선 영* , 박 창 섭*

Link-State Routing Security Mechanism based on Double Hash Chain

Byung-Ik Yoo*, Jung-Mi Lim*, Sun-Young Yoo*, Chang-Seop Park*

요 약

인터넷 상에서의 보안문제는 거의 대부분이 사용자 데이터에 대한 보안에 초점이 맞추어져 있는 반면에, 인터넷의 원활한 운영에 중요한 골격을 이루는 라우팅 프로토콜의 안전성에 대한 연구는 그리 폭 넓게 행해지고 있지는 않다. 본 논문에서는 현재 인터넷에서 가장 보편적으로 사용되고 있는 Link-State 라우팅 프로토콜의 보안상의 문제점을 지적하고, 이를 위해서 제안된 기존의 일부 인증 메커니즘들의 기능 및 성능을 보완, 확장한 새로운 라우팅 메시지 인증 메커니즘을 제안한다. 이를 위해서 안전성 증명이 가능한 이중 해쉬체인의 개념을 도입하고 이에 기반을 둔 세션 해쉬체인을 통해서 라우팅 메시지에 대한 무결성 및 근원지 인증 서비스를 제공하는 방안을 소개한다.

ABSTRACT

The current security issue for the Internet is focused on the security for user data. On the other hand, the research on the security for routing protocols is not so active, considering the importance of its role for the harmonious and accurate operation of the Internet. In this paper, we investigate the security problems of the link-state routing protocol which has been employed in the Internet, and suggest a new authentication mechanism for routing messages which complements and extends the previous ones. For this purpose, a concept of dual hash chains is newly introduced, which is provably secure, and we explain how to provide both the integrity and source authentication service for routing messages based on the session hash chains.

Keyword : *multicast routing protocol, hash chain, authentication mechanism*

Ⅰ. 서 론

라우팅 프로토콜(routing protocol)을 통해서 라우터(router)들은 네트워크의 토폴로지(topology)에 관한 정보를 서로 교환하고, 이를 기반으로 근원지에서 목적지로의 패킷(packet) 포워딩(forwarding)을 수행한다. 정확치 않은 라우팅 정보의 교환은 인터넷의 기능을 비효율적으로 저하시킬 수 있으며, 최악의 경우에는 그 기능을 완전히 상실할 수도 있게 된다. 이렇듯 라우팅 프로토콜이 인터넷 하부구조에 있어서 중추적인 역할을 함에도 불구하고, 라우팅 프로토콜은 의도적으로 또는 부주의로 정확치 않은 라우팅 정보가 유포되는 것에 대한 효율적인 방비책을 마련하고 있지 못하고 있다. 그 이유는 기본적으로 라우팅 프로토콜 자체는 TCP 그리고 IP 프로토콜과 마찬가지로, 프로토콜 당사자들이 올바르게 프로토콜의 정해진 작업을 수행한다는 상호간의 묵시적인 신뢰를 바탕으로 설계가 되었기 때문이다. 궁극적으로 라

* 본 연구는 2002학년도 단국대학교 대학연구비의 지원으로 연구되었습니다.
* 단국대학교 전자계산학과

우팅 프로토콜은 라우터 상호간에 교환되는 정보를 기반으로 작동을 하기 때문에, 라우팅 프로토콜을 통해서 전달되는 라우팅 정보는 여러 유형의 공격에 노출될 수밖에 없다.

부정확한 라우팅 정보의 근원지는 크게 2가지로 집약된다. 첫째는 외부 공격자가 라우터와 라우터 사이의 링크 구간에의 접근이 허용되어, 이를 기반으로 전송되는 라우팅 정보의 변조 및 삭제, 그리고 이전에 전송되었던 라우팅 정보의 재생과 위조된 라우팅 정보를 삽입하는 경우이다. 둘째는 라우팅 프로토콜에 참여하는 내부 라우터의 부주의한 구성작업으로 인한 오 작동(malfunction)과 의도적인 구성작업에 의한 오 작동에 기인할 수가 있다. 특히 이 경우, 정상적으로 작동하지 않는 라우터에 의해서 발생되는 부정확한 라우팅 정보는 일시적으로 또는 지속적으로 네트워크의 안정성(stability)을 교란시킬 수가 있다. 따라서, 그러한 라우터를 탐지하여 격리시킬 수 있는 방안도 마련되어야 한다.

현재 인터넷에서 가장 보편적으로 이용되고 있는 내부 라우팅 프로토콜(interior routing protocol)은 크게 2가지로 대별된다. 첫째는 RIP(Routing Information Protocol)[1]와 같은 Distance-Vector 라우팅 프로토콜이고, 둘째는 OSPF(Open Shortest-Path First)[2]와 같은 Link-State 라우팅 프로토콜이다. 이 2가지 유형의 라우팅 프로토콜을 비교할 때, 후자의 경우가 전자의 경우보다 여러 측면에서 보다 효율적인 것으로 알려져 있다. 전자의 경우에는 네트워크에 어떠한 변화가 발생되어지지 않았음에도 불구하고 라우터간에 정기적으로 전체 라우팅 테이블을 교환함으로써 불필요한 네트워크의 트래픽을 증대시킨다. 하지만, 후자의 경우에는 단지 네트워크에 특정 변화가 발생할 경우에만 전체 라우팅 테이블이 아닌, 해당 변화된 사항만을 선별적으로 교환하여 네트워크의 효율성을 유지할 수 있게 한다. 또한, 전자의 경우에는 목적지까지의 최적경로(best route)를 계산함에 있어서 단지 hop의 개념만을 사용하지만, 후자에서는 hop 뿐만 아니라 대역폭(bandwidth), 지연(delay), 신뢰성(reliability) 등의 여러 metric을 이용하여 최적경로를 계산한다. 본 논문의 기본적인 목적은 Link-State 라우팅 프로토콜을 대상으로 상호 교환되는 라우팅 정보에 대한 무결성(integrity) 및 근원지 인증(source authentication) 서비스를 제공하기 위한 방안을 제시하는 데에 있다. 2장에서는 Link-State 라우팅 프로토콜에 대한 개략적인 설명, 그리고 기존에 제안된 보안 메커니즘에 대한 연구를 소개한다. 3장에서는 기존 방식의 문제점을 보완하여 확장한 인증 메커니즘을 제안하고, 4장에서는 기존 방식과의 효율성측면에서의 비교분석을 한다.

II. Link-State 라우팅 프로토콜 보안

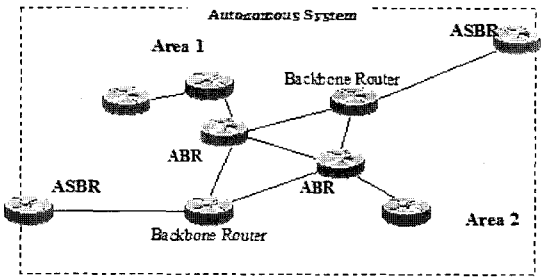
이번 장에서는 Link-State 라우팅 프로토콜에 대한 개략적인 내용을 기술하고, 또한 Link-State 라우팅 프로토콜에 적용 가능한 기존의 라우팅 보안 메커니즘들에 대한 관련 연구를 소개한다.

2.1 Link-State 라우팅 프로토콜

라우팅 관점에서 보면, 인터넷은 AS(Autonomous System)라고 하는 독립적인 영역으로 분할된다. AS는 관리적인 측면에서 한 단체에 속하여 관리되고 제어되는 동일한 라우팅 정책을 사용하는 네트워크 그룹을 지칭한다. 따라서, 특정 AS 내에서는 통일된 내부 라우팅 프로토콜(interior routing protocol)을 통해서 정보가 교환되는데, 가장 보편적으로 사용되는 내부 라우팅 프로토콜로는 OSPF[2]를 들 수 있다.

각각의 AS는 네트워크 및 라우터의 개수가 증대되어짐에 따라서 한 개 이상의 라우팅 도메인(routing domain), 즉 Area로 분할되어질 수 있다. [그림 1]의 예에서는 AS가 모두 3개의 Area, 즉 Area 1, Area 2 그리고 나머지 Backbone Area로 구성이 되어져 있다. 2개 이상의 Area에 속해 있는 라우터를 ABR(Area Border Router)이라고 하며, 다른 AS와의 연결을 위해서 존재하는 라우터를 ASBR(Autonomous System Boundary Router)이라고 한다.

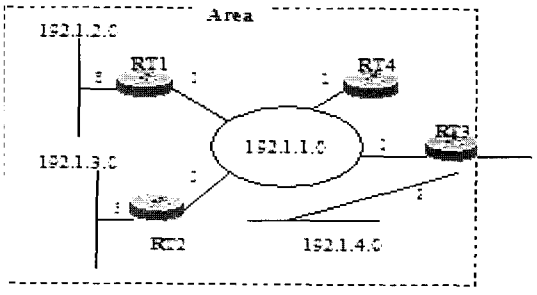
특정 Area에 속해 있는 내부 라우터들은 해당 Area 내에서 자신과 직접 연결되어 있는 네트워크 인터페이스 또는 다른 내부 라우터에 대한 정보를 해당 Area에 존재하는 다른 라우터들에게 플러딩(flooding) 기법을 통해서 전달한다. 각각의 라우터가 다른 라우터들에게 플러딩 시키는 라우팅 정보는 LSA(Link State Advertisement) 패킷(packet)을 이용하게 된다. 결과적으로 특정 Area에 속해있는 내부 라우터들은 그 Area에 대한 동일한 네트워크 토폴로지를 유지할 수가 있다. 이 네트워크 토폴로지를 Link-State 데이터베이스라고도 하며, 결국 이를 기반으로 각각의 라우터들은 자신의 라우팅 테이블을 구축한다. ABR과



〔그림 1〕 Autonomous System

ASBR은 각각 한 개 이상의 Area와 AS에 걸쳐 있기 때문에, 다른 Area나 AS에의 라우팅 정보를 내부 라우터들에게 전달해 주는 역할을 한다.

[그림 2]는 특정 Area 내의 네트워크 및 라우터 구성을 보여주고 있다. 여기서 RT3는 해당 Area와 Backbone Area를 연결시켜주는 ABR이고, 네트워크 192.1.2.0, 192.1.3.0, 192.1.4.0은 Stub 네트워크, 그리고 192.1.1.0은 Transit 네트워크이다. Transit 네트워크는 인접한 2개의 네트워크 사이에 전달되는 패킷을 통과시킬 수 있는 네트워크를 의미하며, Stub 네트워크는 오직 1개의 네트워크에만 연결이 되어 있어서 패킷을 통과시키는 기능은 가지고 있지 못하다. 각 링크 위에 표시된 숫자는 metric, 즉 링크의 상태 값을 지칭하며 최종 목적지까지 패킷이 도달하기 위한 최적 경로를 계산하기 위해서 사용되는 cost이다. 예를 들어, RT3가 해당 Area에 플러딩 시키는 LSA에는 RT3가 네트워크 192.1.4.0과 192.1.1.0에 연결되어져 있고, 해당 링크 상태 값은 각각 1과 2라는 정보가 포함된다. 이와 같이 RT1, RT2, RT3, RT4는 LSA의 교환을 통해서 해당 Area에 대한 동일한 Link-State 데이터베이스를 구축하게 되는데, LSA는 링크 상태의 변화가 있을 때마다 해당 Area에 플러딩 된다.



〔그림 2〕 Area 내의 구성 예

2.2 관련 보안 메커니즘 연구

라우팅 프로토콜의 안전하고 정상적인 동작을 보장하기 위한 보안 메커니즘에 대한 기존의 연구는 라우팅 메시지에 대한 무결성 및 근원지 인증을 제공하기 위한 것이다. 이에 대한 가장 선도적인 연구는 Perlman[3], 그리고 Murphy와 Badger[4]에 의해 행해졌다. 이들은 디지털 서명을 통한 라우팅 정보에 대한 무결성 그리고 근원지 인증을 제공하는 방안을 제시하였다. 특히, Murphy와 Badger[4]는 OSPF 라우팅 프로토콜에 구체적으로 적용할 수 있는 세부적인 인증 메커니즘을 제안하는데, OSPF에 대해서 가해질 수 있는 다양한 공격의 유형과 그에 따른 효과를 잘 설명하고 있다. 실제로 OSPF에는 자체적으로 인증 서비스를 제공하기 위한 메커니즘이 마련되어 있지만, 그것은 라우터 간에 패스워드를 공유하거나, 또는 비밀키 공유를 기반으로 MAC(Message Authentication Code) 값을 계산하여 무결성 및 인증 서비스를 제공하자는 내용이다. 하지만, 평문의 형태로 교환되는 패스워드의 보안성은 의미가 없으며, 또한 비밀키 기반의 MAC도 라우터가 다른 모든 라우터와의 서로 다른 비밀키를 공유하는 것을 전제로 하고 있기 때문에, 라우팅 도메인이 매우 큰 시스템에서는 운영상의 문제점이 많아 실효성이 없다.

디지털 서명에 기반을 둔 인증 메커니즘의 경우, 모든 라우팅 정보에 대해서 서명을 하고, 또한 이를 수신한 라우터들이 그 서명을 확인하는 작업에는 많은 시간적, 계산적 부담이 초래된다. 특히, Murphy와 Badger[4]의 제안은 IETF meeting에서 표준안으로 상정되었으나 이러한 이유로 인하여 그 안이 부결되었다. 이에 착안하여 Hauser 등[5]은 해쉬체인에 기반을 둔 인증 메커니즘을 제안하였다. 매 세션마다 생성되는 라우팅 정보를 디지털 서명하는 대신에 단지 최초의 세션에 대해서만 디지털 서명을 요구하고, 그 이후의 세션들에서는 일방향 해쉬함수에 기반한 해쉬체인이 사용된다. 매 세션마다 해당 링크가 연결되어있는지 또는 연결되어있지 않은지 만을 나타내는 "UP" 과 "DOWN" 의 2가지 상태를 표현하기 위해 2개의 독립적인 해쉬체인을 통한 라우팅 정보의 무결성을 보장하고 있다. 특정 링크가 "UP" 일 경우에는 "UP" 에 대한 해쉬체인의 다음 값을 플러딩 하고, 반대로 "DOWN" 일 경우는 이에 대응되는 해쉬체인의 다음 값을 역시 플러딩 한다. 해쉬함수 계산이 디지털 서명 및 확인에 소요되는 계산 복

잡도보다 훨씬 덜하다는 장점이 있지만, 그들의 제안은 기본적으로 링크의 상태 값 표현에 한계를 가지고 있다. 즉, 해당 링크의 2가지 상태만을 나타낼 수 있는 방안이기 때문에, 현실적으로 여러 링크 상태 값을 표현하기 위해서는 효율적인 대안이라고는 할 수 없다. 이와 관련하여 본 논문의 직접적인 동기는 Hauser 등[5]이 제안한 방식을 보완할 수 있는 방안을 찾고자 하는 데에 있다.

Cheng[6]은 OSPF에 내재된 MAC기반의 인증 메커니즘을 변형한 방안을 제시하였다. 그의 제안은 MAC에 소요되는 비밀키를 생성하고 그것을 서로 다른 한 쌍의 라우터에게 분배, 관리하는 데에는 문제점이 많다는 데에 기인한다. 즉, 라우팅 정보를 송신하는 라우터가 자신이 임의로 생성한 비밀키로 먼저 MAC을 계산하여 수신자 라우터에게 보내고, 나중에 그 MAC을 확인할 수 있는 비밀키를 보내는 방안이다. 이 방식의 특징은 매 세션마다 소요되는 연속적인 비밀키의 생성을 위해서 일방향 해쉬함수에 기반한 해쉬체인을 이용하는 데에 있다. 하지만, 비밀키의 사전 분배 문제점을 해소했다는 측면에서는 의미가 있지만, 해당 라우팅 정보를 수신한 라우터가 그것에 대한 인증 확인작업이 없이 그 정보를 자신의 라우팅 테이블을 작성하는 데에 이미 반영했다는 데에 또 다른 문제점을 수반하고 있다. 만약, 그 해당 라우팅 정보에 대한 인증이 실패할 경우에는 해당 라우터는 다시 이미 작성한 라우팅 테이블을 보정해야 하는 추가적인 작업 부담을 가지게 된다. Goodrich[7] 역시 비밀키 기반의 MAC을 이용한 인증 메커니즘을 제안하였으나 그 역시 비밀키 사전 분배에 대한 방안은 제시하지 못하고 있다.

Zhang[8]은 Murphy와 Badger[4]의 디지털 서명방식과 Hauser 등[5]의 해쉬체인 방식의 문제점을 보완하여 해쉬체인에 기반을 둔 일회용 디지털 서명(One-Time Signature) 방식의 인증 메커니즘을 제안하였다. 하지만, 그의 제안은 라우팅 정보에 해쉬함수를 적용해서 나온 결과 값의 각각의 비트에 대해서 일회용 서명을 하는 방식이기 때문에 서명의 길이가 너무 커진다는 단점을 가지고 있다.

Ⅲ. 새로운 라우팅 메시지 인증 메커니즘의 제안

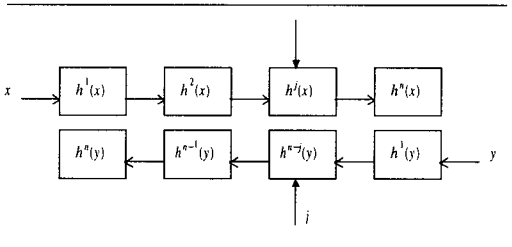
이번 장에서는 한 개의 Area 내에서 교환되는 라우팅 정보, 즉 LSA(Link State Advertisement)를 보호하기 위해 제안되었던 디지털 서명방식과 해쉬체인에 기반을 둔 방식의 단점을 보완 및 확장한 인증 메커니즘을 제안한다. 이를 위해서 본 논문의 핵심이 되는 이중 해쉬체인의 개념을 먼저 정의하고, 이에 대한 안전성을 증명한다.

3.1 이중 해쉬체인

일반적으로 단일 해쉬체인(hash chain)은 seed 값 x가 주어졌을 때, 이를 기반으로 반복적인 일방향 해쉬함수 h의 적용을 통해서 나오는 값들의 체인 $h^1(x), h^2(x), \cdots, h^n(x)$를 의미한다. 여기에서 $h^i(x)$의 의미는 x에 해쉬함수를 i번 반복 적용한 결과 값이다. $h^i(x)$로부터 $h^{i+1}(x)$의 값은 계산할 수 있으나, $h^{i-1}(x)$는 도출할 수가 없는 일방향 해쉬함수의 특성 때문에 해쉬체인은 소액 전자현금이나 일회용 패스워드 메커니즘에 이용되어 왔다. 본 논문에서 제시하는 이중 해쉬체인은 단일 해쉬체인과는 달리, 특정 해쉬체인의 값이 주어졌을 때 양방향 모두 계산이 불가능하게 함으로써 특정 해쉬체인 값의 무결성을 보장하기 위한 목적으로 제안된다. 본 논문에서는 링크 상태 값이 이중 해쉬체인 상에서의 특정 해쉬 값의 위치로 표현된다.

2개의 seed 값 x와 y를 임의로 선정하고, 일방향 해쉬함수 h를 이용하여 다음과 같이 길이가 n인 2개의 해쉬체인을 생성한다.

이때 $h^n(x) \oplus h^n(y) = v$를 이중 해쉬체인 확인자(double hash chain verifier)로 정의한다. 이 해쉬체인을 작성한 사용자는 확인자 v에 대해서 자신의 서명용 개인키를 이용하여 디지털 서명한 값을 다른 사용자에게 보내고, 서명을 수신한 사용자는 그 서명을 작성자의 공개키를 이용하여 확인한 후에, 그 값을 저장한다. 차후에, 그 해쉬체인의 작성자는 임의의 $j \in [1, n-1]$ 값을 다음과 같이 무결성이 보장되게 보낼 수가 있게 된다.



〔그림 3〕 이중 해쉬체인

$$\{ j,\ h^j(x),\ h^{n-j}(y)\ \}$$

이 메시지의 수신자는 $h^j(x)$와 $h^{n-j}(y)$에, 각각 $n-j$번과 j 번의 해쉬함수를 반복 적용하여 나온 결과값 $h^n(x)$와 $h^n(y)$를 XOR한 값과, 자신이 저장하고 있던 확인자 v가 일치하는지를 확인하게 된다. 여기에서 하나가 아닌 2개의 해쉬체인을 생성하는 이유는, 하나의 해쉬체인을 사용하는 경우에는 무결성을 위협하는 공격이 가해질 수 있기 때문이다. 즉, 하나의 해쉬체인을 사용할 경우, 메시지 $\{ n-j,\ h^j(x)\ \}$를 전송하게 되면, h^{j+1}이 $h^j(x)$로부터 쉽게 계산되어질 수가 있기 때문에 공격자는 $\{ n-j,\ h^j(x)\ \}$을 $\{ n-j-1,\ h^{j+1}(x)\ \}$로 변조하여 보냄으로써 j 대신 $j+1$ 의 값을 보낸 효과를 얻을 수가 있게 된다. 따라서, 무결성이 보장되지 않는다. 하지만, 본 절에서 제시한 이중 해쉬체인(dual hash chain)의 경우에는 [정리 1]에 나타나 있는 것처럼 그러한 변조가 계산적으로 용이하지가 않게 된다.

[정리 1]

충돌회피 해쉬함수 $h:\{0,1\}^* \rightarrow \{0,1\}^m$이 주어졌을 때, 공격자가 $1/2^m$을 넘는 확률로 j를 $j'(\neq j)$으로 변경하고 무결성 검사를 통과하는 것은 계산적으로 어렵다.

(증명)

공격자에게 주어진 정보는 현재 전송중인 *message* = $\{ j,\ h^j(x),\ h^{n-j}(y)\ \}$와 그것으로부터 계산되어질 수 있는 값 즉 이중 해쉬체인 확인자 $v = h^n(x) \oplus h^n(y)$ 이다. 먼저, $j'(>j)$인 경우에 공격자는 $h^j(x)$에 $j'-j$ 번의 해쉬함수를 반복 적용하여 $h^{j'}(x)$을 구할 수가 있다. 만약, 공격자가 $h^{n-j'}(y)$을 구할 수만 있다면 *message*를 *message'* = $\{ j', h^{j'}(x), h^{n-j'}(y)\ \}$으로 변조하여도 수신측 무결성 검사를 통과할 수가 있다. 하지만, 이는 충돌회피 함수의 특성상 계산적으로 어렵다. 따라서, 공격자는 $z = h^{n-j'}(y)$을 만족하는 임의의 $z \in \{0,1\}^m$를 선택할 수 밖에는 없고, 결과적으로 성공할 확률은 $1/2^m$ 을 넘을 수가 없다. $j'(<j)$인 경우에도 동일한 결과를 얻게 된다.

이와 같이 이중 해쉬체인 기법을 이용하면 일정한 범위 내에 있는 값 $j \in [1, n-1]$를 메시지에 대한 무결성 및 근원지 인증(source authentication)이 보장되게 전송할 수가 있다. 이중 해쉬체인의 사용은 일회용이다.

3.2 인증 메커니즘

이번 절에서는 이중 해쉬체인 기법을 이용하여 특정 라우터에 존재하는 $k(\geq 1)$개의 링크 상태(link state)에 대한 정보를 무결성 및 근원지 인증이 보장되게 해당 Area의 다른 라우터들에게 플러딩 시키는 인증 메커니즘에 대하여 논의한다. 먼저, 이전의 제안들에서처럼[3,4,5,7], 해당 Area에 적용되는 공개키 기반구조가 존재한다는 가정을 한다. 즉, AS 내에는 인증기관의 역할을 하는 서버가 존재하며 특정 Area에 새로이 참가하는 모든 라우터는 자신의 서명용 개인키와 이에 대응되는 공개키 인증서와 더불어 인증기관의 공개키를 부여받게 된다.

3.2.1 시스템 생성

라우터는 인접하는 각각의 링크 $L_l(1 \leq l \leq k)$마다 임의의 seed값 $x_1^{(l)}$ 과 $y_1^{(l)}$ 그리고 일방향 해쉬함수 h를 이용하여 길이가 n인 "이중 해쉬체인"을 생성하고, 해쉬체인의 길이 n은 라우터와 인접한 링크의 상태 값 $j \in [1, n-1]$의 개수를 고려하여 설정한다. 여기에서 $n-1$은 링크의 DOWN을 의미하며, 1부터 $n-2$까지의 값이 UP 상태에 있는 링크의 다양한 상태 값을 표시하게 된다.

라우터가 LSA 패킷을 통해서 해당 Area에 플러딩 시키는 링크 상태 값은 매 세션마다 갱신되어질 수가 있기 때문에, $g(a,b) = g(a \| b)$와 같이 정의되는 일방향 해쉬함수 $g:\{0,1\}^m \times \{0,1\}^m \rightarrow \{0,1\}^m$에 기반을 둔 다음과 같은 세션 해쉬체인(session hash chain)을 정의한다. $g_0^{(l)} = IV^{(l)}$를 초기화 벡터 그리고 $x_1^{(l)}$ 와 $y_1^{(l)}$을 기반으로 하여 매 세션마다 이중 해쉬체인에 사용될 seed 값을 $x_{i+1}^{(l)} = x_i^{(l)} + 1$와 $y_{i+1}^{(l)} = y_i^{(l)} + 1$라고 하자. 이때, $l = 1,2,\ldots,k$ 그리고 $i = 1,2,\ldots,t$에서 k는 링크의 개수, t 는 정의되는 세션 해쉬체인의 최대 길이가 된다. 매 세션마다 이와 같이 seed 값을 생성하는 이유는 seed 값에 대한 저장공간을 줄이기 위해서 이다.

위에서 $v_{t-i+1}^{(l)} = h^n(x_i^{(l)}) \oplus h^n(y_i^{(l)})$는 $i(= 1,2,3,..)$번째 세션에 링크 L_l에 적용되는 이중 해쉬체인 확인자를 의미한다. 라우터 RT는 자신이 위치하고 있는 Area의 라우터들에게 다음과 같이 자신의 개인키 SK로 디지털 서명이 된 세션 초기값을 플러딩 한다. 세션 초기값에는 현재의 시각표(Timestamp) T_0와 초기 순번(sequence number) SN_0, 그리고 각 링크에 대해서 설정된 세션 해쉬체인의 확인자(session

hash chain verifier) $g_t^{(1)}, g_t^{(2)}, \ldots, g_t^{(k)}$가 포함된다. 또한 세션 초기값과 함께 자신의 공개키 인증서 $Cert_{RT}$도 함께 보낸다.

$$[RT, T_0, SN_0, g_t^{(1)}, g_t^{(2)}, \ldots, g_t^{(k)}]_{SK}, \ Cert_{RT}$$

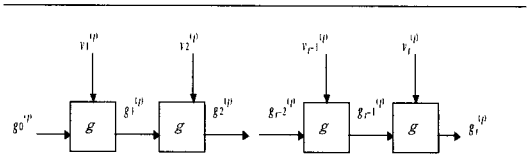
위의 메시지를 전달받는 라우터들은 $Cert_{RT}$를 이용하여 서명의 유효성을 확인한 후에, 세션 초기값을 저장한다. 이와 같이 디지털 서명은 세션 초기값을 전달할 경우에만 사용된다.

3.2.2 링크 상태 패킷의 생성 및 처리

i (=1,2,3,...) 번째의 세션에 해당하는 링크 상태 패킷 LSA는 다음과 같이 작성되어 라우터들에게 플러딩 된다. 즉, 각각의 링크 L_l 에 대한 상태값을 $j_i^{(l)}$이라고 하고, k개의 이중 해쉬체인 $LS_i^{(l)} = \{j_i^{(l)}, h^{j_i^{(l)}}(x_i^{(l)}), h^{n-j_i^{(l)}}(y_i^{(l)})\}$for $l = 1,2,\ldots,k$을 계산한다. 그리고 다음과 같은 링크 상태 패킷 LSA_i를 생성하여 플러딩 한다.

위의 링크 상태 패킷을 수신한 각각의 라우터는 그 패킷에 대한 무결성 및 근원지 인증을 확인하기 위해, 우선 LSA_i안의 T_i가 갱신된 것이고 또한 SN_i이 현재 예상된 순번일 경우에 한해서 LSA_i 에 포함되어 있는 각각의 $\{ g_{t-i}^{(l)}, LS_i^{(l)}\}$에 대해서 [그림 6]에서와 같은 계산을 한다. 즉, $LS_i^{(l)}$에 있는 $h^{j_i^{(l)}}(x_i^{(l)})$과 $h^{n-j_i^{(l)}}(y_i^{(l)})$에 대해서 각각 $n-j_i^{(l)}$번과 $j_i^{(l)}$번의 해쉬함수를 반복 적용하여 나온 결과값을 기반으로 이중 해쉬체인 확인자 $v_{t-i+1}^{(l)} = h^n(x_i^{(l)}) \oplus h^n(y_i^{(l)})$를 구성하고, 함께 수신된 $g_{t-i}^{(l)}$를 이용하여 계산된 $g(g_{t-i}^{(l)}, v_{t-i+1}^{(l)})$ 값이 이전 세션에서 수신되어 저장된 "세션 해쉬체인 확인자" $g_{t-i+1}^{(l)}$과 일치하는지 확인한다. 성공적으로 확인이 되면 라우터는 자신의 Link-State 데이터베이스에 해당 링크 상태 값들을 수신된 새로운 값으로 갱신한다.

Link-State 라우팅 프로토콜은IP 계층 바로 위에서 작동되기 때문에 TCP의 신뢰성 서비스를 직접적으로 제공받지는 못한다. 하지만, 자체적으로 운영하는 신뢰성 메커니즘이 이를 보완하고 있다. 즉, 플러딩되는 링크 상태 패킷을 수신한 라우터는 해당 패킷을 송신한 라우터에게 Acknowledgement 패킷을 전송함으로써 해당 링크 상태 패킷의 성공적인 수신을 알리게 된다. 링크 상태 패킷의 안정적인 전달을 위해



〔그림 4〕 세션 해쉬체인

서 송신측 라우터는 수신측으로부터 Acknowledgement 패킷이 수신될 때까지 동일한 링크 상태 패킷을 사전에 구성된 시간 간격을 두고 반복해서 전송한다.

3.2.3 복구 메커니즘

수신된 링크 상태 패킷 LSA에 대한 인증이 실패하는 경우는 외부 공격자 또는 LSA가 플러딩 되어지는 과정에 참여하는 일부 라우터의 잘못된 구성작업 및 공격으로 인하여 패킷이 변조되는 경우이다. 또한, 이들에 의해서 특정 LSA가 임의로 생성되어지거나 삭제되어질 수도 있다. 실제로 Link-State 라우팅 프로토콜[2]에서는 이렇게 위조 또는 변조된 LSA가 해당 Area 내에서 플러딩 되어지기 때문에 궁극적으로 원래 그 LSA를 최초로 생성했던 Advertising Router에게도 도달되어지게 된다. 따라서, 그 Advertising Router는 이를 보정하는 일종의 "fight-back" LSA를 다시 플러딩하는 방법을 취하게 된다. 하지만, 이 방식은 전체 네트워크의 안정화에 많은 시간적 지연이 초래되어지는 단점을 가지고 있다.

본 논문에서 제안하는 인증 메커니즘과 연계된 복구 메커니즘은 [그림 7]과 같이 작동한다. 즉, 라우터 RT1으로부터 받은 LSA가 인증에 실패할 경우에는 라우터 RT2는 현재 수신한 LSA를 더 이상 인접한 라우터들에게 플러딩 시키지 않고, 원래 그 LSA의 Advertising router인 AR에게 유니캐스트로 자신이 수신한 LSA를 보내어 확인을 요청하게 된다. 이때, 라우터 RT2의 서명용 개인키로 서명을 해서 보낸다. 확인 요청 메시지를 수신한 라우터 AR은 해당 링크 상태 메시지를 자신의 개인키를 이용하여 서명을 하고 그 결과를 다시 요청 라우터인 RT2에게 유니캐스트로 전달해 준다. 서명을 확인한 라우터 RT2는 자신의 Link-State 데이터베이스를 갱신하고, 인접한 라우터들에게 해당 LSA를 플러딩 한다.

3.3 안전성 분석

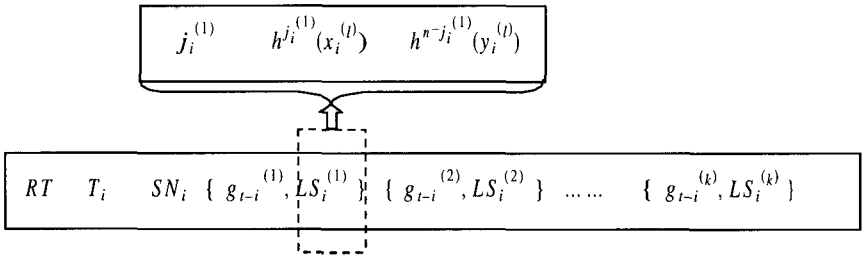
안전성 분석에 있어서는 외부 공격자에 의한 링크

상태 패킷에 대한 변조 및 위조 공격, 그리고 라우팅에 참여하는 라우터들이 비정상적으로 작동을 하여 정상적인 링크 상태 패킷을 변조하는 경우를 대상으로 한다. 기존에 해쉬체인을 기반으로 제안된 메커니즘[5,6,7]의 경우, 해당 Area에 존재하는 라우터들 사이에 시간에 대한 엄격한 동기화(clock synchronization)가 이루어지지 않을 경우에는 "지연-위조" (delay-and-forge) 공격이 가능하게 된다. 즉, 시각 $T_{i_1} < T_{i_2}$의 경우에 세션 i_1에서 보낸 메시지를 이용하여 세션 i_2에 그것을 그대로 또는 변조하여 보내는 공격을 의미한다. 따라서, 만약 그 메시지를 수신하게 되는 라우터가 송신 라우터와의 시각 동기화에 현격한 차이가 나고 그리고 세션 i_1에서 보낸 메시지를 아직 받지 못한 상황에서는 위의 공격이 성공할 수 있게 된다. [5,6,7]들에 대해서 "지연-위조" 공격이 가능하게 되는 근본적인 원인은 공격자가 특정 시점까지 관측한 메시지들을 기반으로 하여 변조된 또는 순서가 바뀐 메시지를 구성하는 것이 가능한 구조를 지니고 있기 때문이다.

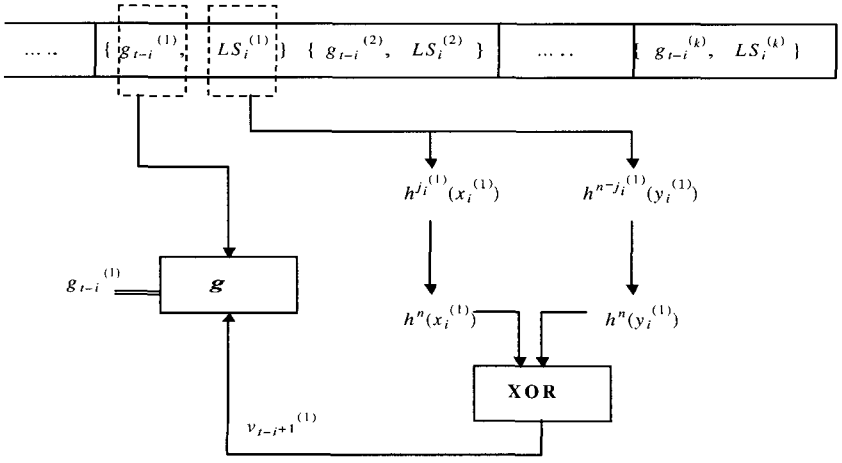
본 논문에서 제안하는 방식은 [정리 2]에 나타나 있는 것처럼 특정 시점까지의 모든 메시지들이 관측이 되었다고 할지라도 그것으로부터 그 이전 또는 그 이후에 유효하게 사용이 가능한 변조된 링크 상태 패킷을 구성하는 것은 불가능하다.

[정리 2]

공격자에 의서 $w(\geq 1)$ 개의 연속 된 링크 상태 패킷, 즉 $LSA_{i_1}, LSA_{i_2}, \ldots, LSA_{i_w}$, $(i_1 < i_2 < \ldots < i_w)$ 가 관측되었다고 하자. 이때, 공격자가 이를 기반으로 과거 또는 미래의 특정 링크의 상태 값을 변조시킬 수 있는 정당한 링크 상태 패킷을 도출 해 내는 것은 3 불가능하다.



〔그림 5〕 LSA 패킷의 구성



〔그림 6〕 LSA 패킷의 처리

(증명)

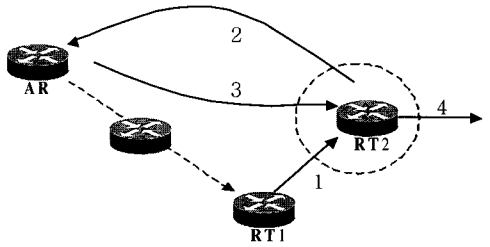
공격의 대상이 되는 특정 라우터가 현재 저장하고 있는 "세션 해쉬체인 확인자"를 $g_{t-i}^{(l)}$ 그리고, $i+1=i_1$ 라고 하자. 이는 결국 현재 해당 라우터가 이미 전송된 w개의 링크 상태 패킷을 수신 받지 못한 상황이다. 공격자의 목적은 이중 해쉬체인 값 $LS_{i_1}^{(l)}$을 임의로 변조하는 것인데, 이를 위해서는 수신자 라우터가 행하는 계산 $g(g_{t-i_1}^{(l)}, v_{t-i_1}^{(l)}) = g_{t-i}^{(l)}$을 만족시켜야 한다. 이 경우에 2가지를 생각할 수 있다. 첫째는 $LS_{i_1}^{(l)}$로부터 라우터가 계산한 "이중 해쉬체인 확인자" $v_{t-i_1}^{(l)}$이 계산될 수 있게 $LS_{i_1}^{(l)}$을 변조하는 것인데 이는 계산적으로 불가능하다는 것을 [정리 1]에서 증명하였다. 둘째는 $LS_{i_1}^{(l)}$을 임의로 변조하여 $g(g_{t-i_1}^{(l)}, v_{t-i_1}^{(l)}) = g_{t-i}^{(l)}$을 만족하는 $g_{t-i_1}^{(l)}$을 선정할 수 있어야 한다. 하지만, 이는 일방향 해쉬함수 g를 기반으로 2^{md} pre-image를 계산하는 작업이기 때문에, 유효한 값을 선정할 확률은 $1/2^m$을 넘을 수가 없게 된다. 따라서, i_1번째의 링크 상태 패킷에 대한 변조가 불가능하기 때문에 $i_2, \ldots, i_w$번째 링크 상태 패킷에 대한 변조 역시 불가능하게 된다. 반대로, 관측된 w개의 링크 상태 패킷을 기반으로 i_w+1번째의 링크 상태 패킷을 조작해 내는 것 역시 불가능하다. 즉, 이 경우는 공격자가 $g(X, Y) = g_{t-i_w}^{(l)}$,을 만족하는 임의의 X와 Y를 선정해야 한다. 이 경우에도 역시 성공 확률은 $1/2^m$을 넘을 수가 없다.

Ⅳ. 고찰 및 비교분석

이번 장에서는 3장에서 제안한 인증 메커니즘을 기반으로 하여 추가적인 보안 문제 및 효율성 문제를 논의한다.

4.1 오 작동 내부 라우터 문제

특정 내부 라우터가 구성작업의 오류나 또는 악의적인 의도로 오 작동을 할 경우에는 그 원인에 상관없이 네트워크의 성능에 치명적인 영향을 미친다. 이는 비록 외부 공격자는 아니지만, 일종의 내부 공격자로 간주할 수가 있다. 오 작동하는 라우터가 자신이 전달받은 LSA를 다른 라우터에 다시 전달할 때, LSA의 내용을 변조할 수가 있다. 이 경우는 [그림 7]에서 언급한 것처럼 변조된 LSA를 전해 받은 라우터가 수행하는 인증작업을 통해서 그 LSA를 최초로



2. AR ← RT2 : [LSA_i]sk_{RT2}, $Cert_{RT2}$
3. AR → RT2 : [LSA_i']sk_{AR}, $Cert_{AR}$

[그림 7] 복구 메커니즘

작성한 소유자 라우터와의 확인작업을 통해서 보정되어질 수가 있다. 하지만, 문제는 소유자 라우터의 ID를 변조하는 경우이다. 이 경우에 수신 라우터가 행하는 변조된 소유자 라우터와의 확인작업은 의미가 없게 된다. 그 소유자 라우터는 확인 요청을 받은 LSA가 자신이 발생시킨 것이 아님을 확인하게 되고, 궁극적으로 요청 라우터는 변조되기 이전의 LSA에 대한 정보를 잃어버리게 된다. 따라서, 네트워크의 안정화에는 시간적 지연이 초래될 수밖에는 없지만, Link-State 라우팅 프로토콜의 플러딩 기법의 특성상 해당 LSA를 수신하지 못했던 라우터들은 그 이후에 발생되는 LSA를 기반으로 한, 보정 작업을 통해서 소유자 라우터와의 인증 상의 동기화를 이룰 수가 있게 된다.

오 작동하는 내부 라우터가 유발시키는 문제점 중의 또 다른 하나는 임의의 정확치 않은 위조 LSA 패킷을 생성하여 유포시키는 경우를 들 수가 있다. 실제 연결되어 있지 않은 네트워크 또는 라우터에 대한 정보의 전송 등을 들 수 있다. 하지만, 이 경우 역시 Link-State 라우팅 프로토콜의 특성상 해당 Area에 존재하는 각각의 라우터들이 자신의 라우팅 테이블을 계산할 때, 오 작동하는 라우터를 네트워크에서 부분적으로 격리시킬 수도 있다. Link-State 데이터베이스에 Dijkstra 알고리즘을 적용하여 최적경로를 계산할 때, 만약 오 작동 라우터가 존재한다고 주장했던 네트워크 또는 라우터 쪽으로부터 오 작동 라우터에 대한 반대 정보를 받지 못한다면 라우팅 테이블의 생성과정에 오 작동 라우터가 제공하는 정보는 사용하지 않게 된다. 즉, 최적경로 설정을 위해서는 제공되는 정보에 대한 이중의 점검작업이 병행되기 때문이다. 하지만, 오 작동 라우터가 행할 수

있는 공격의 유형은 다양할 수가 있기 때문에 이 분야에 대한 연구는 지속적으로 이루어져야 한다고 사료된다.

4.2 효율성 비교분석

이미 언급한 바와 같이, 본 논문의 직접적인 동기 중의 하나는 Hauser 등[5]이 제안한 방식을 확장 시키는 데에 있다. 그들의 제안은 라우터에 연결된 각 링크별로 "UP"과 "DOWN"에 해당하는 2개의 해쉬체인을 생성한 후에, 해당 링크의 상태에 따라서 이 두 해쉬체인의 값을 교차해서 해당 Area의 라우터들에게 플러딩을 시키는 방안이다. 따라서, 그들의 논문에서 지적된 것처럼 이것을 여러 개의 링크 상태를 표현할 수 있게 확장할 경우에는 각 라우터가 유지, 관리해야 하는 메모리의 양은 크게 증대 된다. 예를 들어, 각 라우터에 연결된 링크의 개수가 $k=3$ 개, 서로 다른 링크 상태 값을 $n=30$, 해당 Area에 존재하는 라우터의 개수는 $c=40$개, 그리고 MD5 기반의 해쉬체인을 사용하고 체인의 길이가 $t=1000$이라고 하자. 이때, 각 라우터가 자신이 전송하는 LSA에 대한 무결성을 보장하기 위해서 유지, 관리해야 하는 메모리는 $128\times1000\times3\times30$ = 1.44M Byte이다. 또한, 해당 Area에 존재하는 다른 라우터가 보내오는 LSA를 인증하기 위한 목적으로 유지, 관리해야 하는 메모리는 $40\times128\times3\times30$ ≈ 57.6K Byte가 된다.

본 논문에서 제안하는 방식의 메모리 양을 비교해 보기로 하자. 먼저, 자신이 유포하는 LSA를 위해서, Hauser 등의 방안처럼 각각의 라우터는 자신만의 세션 해쉬체인을 유지하고 있어야 한다. 세션 해쉬체인의 길이가 역시 1000이고, 그리고 그 값을 해당 세션마다 계산하지 않고 모두 미리 계산한 후에 저장해서 사용한다면 $128\times1000\times(k=3)\times2\approx$ 96K Byte가 소요된다. 본 제안에서는 이중 해쉬체인 기법으로 인하여 서로 다른 링크 상태 값을 별도로 고려할 필요가 없다. 마찬가지로, 다른 라우터가 보내는 LSA를 인증하는 데에 사용할 세션 해쉬체인의 현재 값을 유지하고 있어야 한다. 링크 상태 값에 대한 메모리만을 계산할 경우에, 각각의 LS는 128비트 $\times3$, 링크의 개수 $k=3$, 라우터의 개수는 40이기 때문에 $40\times|LSA|=40\times(128\times3+128)\times3\approx7.68$K Byte가 소요된다. 또한, 각 세션마다 링크 상태 값을 계산하기 위한 이중 해쉬체인에 필요한 seed 값이 128비트라면 이를 위해서 모두 128비트$\times2\times(k=3)=96$ Byte가 소요된다. 결론적으로 해쉬체인 값을 미리 계산해서 저장하고 있는 경우를 가정할 경우에도 각각의 라우터가 인증 메커니즘을 수행하기 위해 필요한 메모리는 Hauser 등의 제안과 달리 개략적으로 수십 K Byte에 지나지 않는다.

〔표 1〕 인증 메커니즘 비교

	제안기법	Hauser.et.al[5]	Cheung[6]	Zhang[7]
특징	이중 해쉬체인	단일 해쉬체인	MAC 키의 지연 전달	일회용 서명
서명개수	1	1	1	1
해쉬체인 생성을 위한 메모리	$2\cdot m\cdot t\cdot k$	$m\cdot t\cdot k\cdot n$	$m\cdot t\cdot k$	$m\cdot t\cdot k\cdot(m+\log m+1)$
해쉬체인 확인을 위한 메모리	$2\cdot m\cdot k\cdot c$	$m\cdot k\cdot c\cdot n$	$m\cdot k\cdot c$	$m\cdot k\cdot c\cdot(m+\log m+1)$
무결성 확인을 위한 메모리	해쉬계산 n번	해쉬계산 1번	해쉬계산1번 MAC계산 1번	해쉬계산 $(m+\log m+1)$번
보정작업	불필요	불필요	필요	불필요

(주): t=체인길이, c=라우터수, k=링크수, m=해쉬값 길이(비트), n=링크 상태 개수

[표 1]은 해쉬체인의 개념을 기반으로 한 다른 인증 메커니즘[5,6,7]과의 비교를 보여주고 있다. m비트의 해쉬함수 값을 이용할 경우, 각 라우터 별로 해쉬체인의 생성을 위해 요구되는 메모리 양, 수신된 메시지의 무결성을 확인하기 위해서 소요되는 메모리양과 각 링크 당 계산량을 비교하였다. 또한, Cheung[6]의 경우에는 MAC 확인에 필요한 비밀키의 역할을 하는 해쉬 값이 지연 전달되기 때문에 MAC 확인이 실패할 경우, 이미 갱신된 Link-State 데이터베이스를 다시 갱신하는 추가의 보정작업이 요구된다. 본 제안을 포함한 다른 메커니즘 모두 첫번째 세션에 생성되는 메시지에 대한 한번의 디지털 서명만이 요구된다.

V. 결 론

인터넷 운영에 있어서 가장 중추적인 역할을 담당하고 있는 라우팅 프로토콜은 자체적으로 충분한 보안 메커니즘을 내포하고 있지 못하다. 따라서, 라우팅 프로토콜이 공격자의 대상이 될 경우에 초래되는 혼란은 누구나 쉽게 예상할 수가 있을 것이다. 본 논문에서는 가장 대표적인 내부 라우팅 프로토콜인 Link-State 라우팅 프로토콜에 적용이 가능한 인증 메커니즘을 제안하였다. 기존에 제시되었던 방안 등을 확장, 개선한 내용을 이중 해쉬체인의 개념을 기반으로 하여 설계하였고, 그의 안전성을 증명하였다.

본 논문에서 충분히 언급하지는 못했지만, 오 작동 내부 라우터에서 발생되는 잘못된 라우팅 정보의 체계적인 탐지와 해당 라우터에 대한 색출은 앞으로의 중요한 연구주제가 되리라고 사료된다.

참 고 문 헌

[1] G. Malkin, "RIP Version 2", *RFC 2453*, 1998.

[2] J. Moy, "OSPF Version 2", *RFC 1583, 1994.*

[3] R. Perlman, "Network Layer Protocols with Byzantine Robustness", *Ph.D. Thesis, Department of Electrical Engineering and Computer Science, MIT, Aug. 1988.*

[4] S. Murphy and M. Badger, "Digital Signature Protection of the OSPF Routing Protocol," *In Proc. of the Symposium on Network and Distributed System Security, pp. 93~102, 1996.*

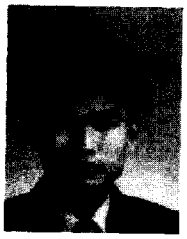
[5] R. Hauser, T. Przygienda, and G.. Tsudik, "Reducing the Cost of Security in Link State Routing", *Computer Networks and ISDN Systems, vol. 31, no. 8, pp. 885~894, 1999.*

[6] S. Cheng, "An Efficient Message Authentication Scheme for Link State Routing," *In Proc. of the 13th Annual Computer Security Applications Conference, San Diego, California, pp. 90~98, 1997.*

[7] K. Zhang, "Efficient Protocols for Signing Routing Messages", *In Symposium on Network and Distributed Systems Security, San Diego, California, 1998.*

[8] M. T. Goodrich, "Efficient and Secure Network Routing Algorithms", *provisional patent filing, U.S.A, 2001.*

〈著者紹介〉



유 병 익 (Byung-Ik Yoo) 준회원
2000년 2월 : 단국대학교 전자계산학과 졸업 학사
2002년 3월~현재 : 단국대학교 전자계산학과 석사과정
<관심분야> 정보보호, 네트워크

임 정 미 (Jeong-Mi Lim) 준회원
2000년 2월 : 단국대학교 전자계산학과 졸업 학사
2002년 2월 : 단국대학교 전자계산학과 석사
2002년 3월~현재 : 단국대학교 전자계산학과 박사과정
<관심분야> 정보보호, 네트워크 보안

유 선 영 (Sun-Young Yoo) 준회원
2002년 2월 : 단국대학교 수학과 졸업 학사
2002년 3월~현재 : 단국대학교 전자계산학과 석사과정
<관심분야> 정보보호

박 창 섭 (Chang-Seop Park) 정회원
1983년 : 연세대학교 경제학과 졸업
1983년 : 한국 IBM 근무
1990년 : 미국 Lehigh Univ. 전자계산학 박사
1990년~현재 : 단국대학교 전자컴퓨터학부 교수
<관심분야> 부호이론, 암호학