主題

도메인네임시스템의 취약점 분석과 보안 확장(DNSSEC)

성균관대학교 김 학 주, 윤 민 우, 임 형 진
한국인터넷정보센터(KRNIC) 송 관 호
성균관대학교 정보통신공학부 정 태 명

차 례

1. 서론

DNS(domain name system)란 IP 주소와 이에 상응하는 계층적 이름 체계를 사상하는 거대한 분산 네이밍 시스템으로써, 인터넷 주소 자원 관리의 핵심이다. 초기 ARPAnet으로부터 시작하여 점차 증가한 인터넷의 규모로 인해, 호스트 이름과 호스트 주소를 사상할 수 있는 기술이 필요했다. 1971년 페기 카프가 HOST. TXT라는 파일 안에 당시 ARPAnet에 연결된 모든 호스트의 이름과 주소를 데이터베이스 형식의 테이블로 저장하고 사상하는 룩업 시스템을 처음으로 고안했다. 이 파일은 당시 SRI-NIC (Stanford Research Institute - Network Information Center)에서 통합 관리했기 때문에 새로운 자원이 네트워크 상에 추가 될 때는 모두 SRI-NIC에 통보해야만 했다. 이러한 구조는 네트워크 규모가 더욱 커지면서 여러 가지 문제를 발생시켰다. 단순한 텍스트 파일 안에 많은 호스트의 이름과 주소를 관리하는 규모의 문제뿐만 아니라 이름 정보에 대한 일관성(consistency)을 보장하기 위한 권한의 문제(problem of authority)가 나타나게 되었다.

이로 인해 새로운 이름-주소 사상 시스템(name to address mapping system)에 대한 요구가 생겼으며 1981년, 데이비드 밀스는 RFC799-Internet Name Domains를 발표한다. 이 문서는 새로운 이름-주소 사상구조인 인터넷 네임 도메인 시스템의 개념을 소개했다. 이어 1982년에는 현재의 DNS의 개념과 구현 방법을 명확히 제시한 RFC882-Domin Names Concepts and Facilities와 RFC883-Domain Names Implementation and Specification이 발표되었다. 이것이 DNS의 시작이다. 두 문서에서 제시하는 가장 중요한 개념은 '위임(delegation)'과 '권한(authority)' 이었다. '권한'은 하나의 개체가 완벽하게 통제 할 수 있는 영향력을 말하며 이 영향력이 인정되는 영역을 존(zone)이라고 한다. 하

나의 존은 주어진 도메인과 그 하부를 이루는 하위 도메인으로 구성된다. '위임'은 어떤 노드에 존에 대한 자신의 권한을 전달해주는 과정으로 설명 될 수 있다. 이 두 가지 정책적 개념을 통해 DNS는 인터넷을 구성하는 각 단위 네트워크의 자유를 최대한 인정하면서, 모든 네트워크 자원의 참조 이름에 대한 일관성과 실시간 업데이트를 보장할 수 있는 계층구조를 갖출 수 있었다. [1][2][19]

인터넷은 기존의 산업을 자신의 영역으로 점차 끌어들이고 있다. 상업과 금융은 이미 e-비지니스에 많은 부분을 의존한다. 인터넷이 제공해야 하는 '신뢰(trustworthy)'는 단순히 기술적 논의의 주제로서만 한정될 수 없다. 인류가 구축한 많은 시스템 자산이 인터넷에 오버레이(overlay)되고 있는 만큼, 인터넷상의 '신뢰'는 그와 같은 자산을 보호해야하는 막중한 책임감을 안고 있다. 인터넷을 구성하는 모든 기반 기술이 이와 같은 '신뢰'를 보장해 줄 수 있어야 하는데 특히 인터넷의 핵심 요소 기술인 DNS는 '신뢰'의 근간이 되는 기반구조 역할을 수행해야만 한다. 최근 인터넷을 기반으로 하는 컴퓨터 네트워크 발전과 더불어 사이버 공격은 해를 거듭할수록 급증하고 있고 그 양상은 악의적인 사용자들에 의한 정보 접근, 정보 조작, 시스템 무기력화 등 고의적이며 불법적인 시도가 주류를 이루고 있다. 또한 2002년 10월에 발생했던 13개의 루트 서버에 대한 서비스 거부공격 등, 인터넷 관리 기반 시설에 대한 공격 또한 증가하고 있어 DNS에 대한 보호가 이슈로 떠오르고 있다.

본 논문은 이와 같은 요구사항을 위한 많은 노력들을 소개하며 DNS의 구조 및 취약성과 최근 이슈가 되고 있는 안전한 DNS(secure-DNS)의 요소기술의 소개, IETF와 해외 연구기관의 DNS 보안 관련 연구 동향 등으로 구성된다.

2. DNS(Domain Name System)

DNS는 트리(tree)형의 분산 데이터베이스(distributed database)이다. 클라이언트(client)와 서버(server) 패러다임으로 구성되어 있으며 트리의 루트 노드(root node)는 "."으로 표시한다. 각 노드는 서브 트리(sub-tree)의 루트 역할을 하며 위임을 통해 권한을 얻은 루트로부터 그 권한이 영향을 미치는 노드들의 영역을 '도메인(domain)이라 하고 각 도메인은 최소한 1개 이상의 네임서버(name server)를 가져야 한다.[1]

2.1 DNS구조

DNS는 크게 리졸버(resolver)와 네임서버로 구성되며 이를 이용해 존과 도메인에 관한 정보를 표현한다. [1]

1) 존 : 존이란 네임 서버에 의해 구분되는 데이터베이스 영역을 말하며 서브 트리(sub-tree)를 구성하는 자원레코드(RR : Resource Record)의 집합이라고 할 수 있다. 존에서는 표현할 데이터를 여러 RR을 통해 구성하는데 여기에는 대표적으로 SOA RR, NS RR, A RR을 사용하여 구역 내의 모든 노드에 대한 신뢰성 있는 데이터, 구역의 상단 노드를 정의하는 데이터, 위임된 하위 존을 설명하는 데이터, 하위 존의 네임서버들에 대한 접근을 허용하는 데이터가 포함된다.
2) 도메인 : 도메인이란 관리의 대상이 되는 개체들의 추상적인 영역을 말한다. DNS에서는 하나의 서브 트리가 하나의 도메인을 이루며 트리 내에서의 등급에 의한 구분과, 위임에 따른 도메인 절단에 의한 구분이 가

능하다.

3) 리졸버 : 리졸버란 호스트 명(host name)을 IP 주소(IP address)로 전환하기 위한 일종의 모듈(module)로서, 명칭 그대로 해결자 역할을 한다. DNS에서 발생하는 질의는 리졸버가 사용자 응용프로그램(user application)으로부터 받아들여 네임 서버로 전송되고, 이에 대한 응답이 리졸버로 수신되면 사용자 응용프로그램으로 전송하게 된다.

4) 네임 서버 : 도메인 데이터베이스를 구성하는 정보의 저장소로써, DNS 요청을 수신하게 되면 동작 방식에 따라 결과를 리졸버에 돌려주는 역할을 한다. 네임서버에서 사용하는 동작방식으로는 비 순환식(non-recursive)과 순환식(recursive)이 있다.

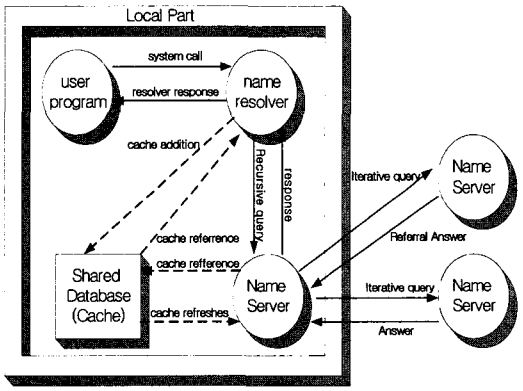
2.2 DNS 구성 및 동작방식

DNS는 리졸버와 네임 서버를 기본으로 하여 캐쉬(cache) 공유 데이터베이스와 외부의 DNS 체계들로 구성된다.([그림 2] 참조) 여기서 외부의 DNS 체계는 DNS 동작에 필요한 요소들이며 내부와 외부의 체계는 연계를 통해 존 트랜스퍼(zone transfer)나 외부로의 요청 및 응답관계가 성립된다.

리졸버는 지역 네임 서버(local name server)와 데이터베이스를 공유하며 캐쉬 된 정보의 내용을 얻거나 캐쉬 정보를 추가하고 지역 네임 서버는 이 데이터베이스를 이용하여 외부로 보낸 질의에 대한 정보를 캐쉬 할 수 있다.

사용자 응용프로그램이 도메인 명을 IP 주소로 전환하기 위한 질의(query)를 리졸버에 전송하면 리졸버는 이 정보를 질의 형식으로 바꾸어 처리를 시작한다. 만약 기존에도 같은 질의가 이루어져 캐쉬 정보가 공유데이터베이스 내에 저장되어 있다면 리졸버는 사용자 응용프로그램에 바로 응답한 후 처리가 끝나며 그렇지 않을 경우, 외부에 있는 네임 서버에 이에 대한 질의를 보낸다. 외부의 네임 서버는 전송 받은 질의에 대해 위에서 설명했던 네임 서버의 동작방식에 따라 처리를 한다. 비 순환식 동작에서는 해당 네임 서버 내에 응답할 수 있는 데이터가 있을 경우 리졸버에 응답을 보내며 그렇지 않을 경우 참조할 수 있는 다른 네임 서버의 주소를 알려 리졸버로부터 재전송된 질의를 통해 최종 응답을 얻어낸다. 순환식 동작 방식에서는 질의를 받은 네임 서버가 다른 네임 서버들로부터 정보를 수집하여 최종 응답을 얻은 후에 리졸버에 그 응답을 보낸다.

네임 서버는 다른 네임 서버들과 정보의 동기화를 위해 유지관리가 필요하다. 여기서 주 네임 서버(primary name server)와 부 네임 서버(secondary name server)의 구분이 이루어지며 SOA RR내에 명시되어 있는 필드들의 값을 이용하여 주기적으로 폴링(polling) 후 주 네임 서버로부터 존 파일(zone file)을 전송받는다. [2]



[그림 2] DNS의 기본 구조

2.3 DNS의 보안 취약성

1990년, AT&T Bell 연구소의 벨로빈은 자신의 논문에서 최초로 DNS의 보안 취약성을 언급

했다. 벨로빈이 분석한 DNS의 보안 취약성은 당시로서는 극복할 수 있는 보안 메커니즘이 없었기 때문에 5년이 지난 1995년이 되서야 발표되었다. 벨로빈의 논문에서 최초로 분석된 DNS의 취약성은 원격사용자가 r-commands를 통해 시스템 접근 시, 호스트 네임 정보를 통한 사용자의 인증 과정에서 발생할 수 있는 버클리 r-commands의 취약성에서 기인한 것이었다. 벨로빈은 rlogin등의 명령어를 사용한 DNS 시스템 공격을 시뮬레이션 했고, 그 결과로서 DNS 데이터베이스에 대한 허가되지 않은 임의의(혹은 악의적인) 변경과 캐쉬 오염 등의 취약성을 도출해 냈다. 최근에는 당시에 문제가 되었던 r-commands는 보안상의 취약성으로 인해 사용하지 않도록 권고되므로, 벨로빈의 분석 자체가 지금 시점에서 중요한 의미를 갖지는 않는다고 할 수 있지만, 벨로빈의 논문을 통해 DNS의 취약성이 갖는 근본적인 문제점이 제시되었다고 할 수 있겠다.[4]

DNS 보안 기술의 개발을 주도하고 있는 IETF의 DNSSEC WG(워킹그룹)에서는, 정보 데이터의 무결성이 보장되지 않는 DNS 환경에서 발생 가능한 취약성을 몇 가지 관점에서 분류하고 있다. <표 1>은 IETF의 취약성 분류를 나타낸 것이다.

서비스 관점에서 DNS의 취약성은 공격자에 의해 위·변조된 정보를 서버가 클라이언트에게 제공함으로서 공격자가 의도한 목적지로 사용자를 유인하거나 (1.b), 서비스거부 공격을 가능하게 한다.(1.a, 1.c)

공격형태의 관점에서는 패킷 가로채기를 변형시킨 다양한 위·변조 공격이 존재한다 (2.a, 2.b). 이를 통해 DNS는 공격자에 의해 변조된 잘못된 응답을 클라이언트에게 보내는 취약점이 발생한다. (2.c)에 기술된 네임기반 공격 방식은 전형적인 DNS 공격 형태중 하나이다. 공격 방법에는 캐쉬 오염(cache poisoning)과 위조된 권한(fake authority)과 같은 형태의 변형들이 존재한다. 이러한 공격의 일반적인 특징은 질의에 대한 응답 메시지가 공격자에 의해 선정된 임의의 DNS 서버를 참조하게 만든다는 것이다.

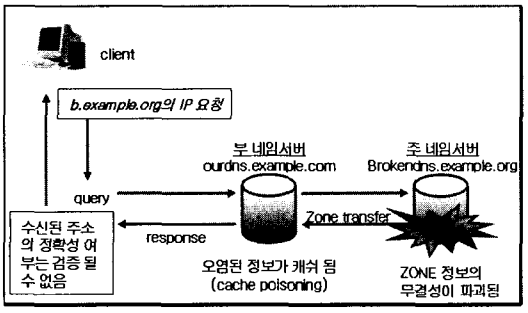
[그림 3]은 주 네임서버에 대해 충분한 제어권을 가진 공격자가 부 네임서버의 캐쉬를 오염시키는 과정을 나타내고 있다. 여기에서 클라이언트는 "b.example.org"의 주소를 요청하고 있다. 이에 대해 응답하는 것은 로드 밸런싱(road balancing)을 위해 주 네임 서버의 작업을 분산하고 있는 부 네임서버("ourdns. example.com")인데, 부 네임 서버는 그림과 같이 공격자에 의해 데이터의 무결성이 침해된 네임서버로부터 잘못된 데이터를 전송받아 캐쉬가 오염된 상태이다. 결국 사용자의 요청에 알맞지 않은 잘못된 주소 정보를 반환하여도 사용자는 이를 검증할 수 있는 메커니즘이 존재하지 않아 공격자의 의도대로 행동하게 되며 네임서버간의 동기화에 따라 주 네임서버뿐만 아니라 DNS를 구성하고 있는 모든 시스템이 오염된 캐쉬의 정보를 사용하게 된다.[6]

패킷 가로채기의 또 다른 변종으로 신뢰된 서버로의 위장을 통한 위·변조 공격(2.d)이 있다. 프로토콜 관점에서 가로채기와의 차이는 클라이언트가 질의를 자발적으로 공격자의 제어 권한 아래에 있는 네임서버에 전송한다는 것이다.

공격 대상 관점에서 살펴보면 DNS 공격자는 호스트 네임 스푸핑(Host Name Spoofing), DNS 스푸핑 (DNS Spoofing)등을 통해서 DNS 클라이언트와 네임서버의 정보에 대한 위·변조 공격을 할 수 있다. 호스트 네임 스푸핑은 DNS 데이터의 PTR RR이 가리키는 주소의 위·변조를 통한 공격으로 호스트 네임 인증을 사용하는 서비스(예. remote login, NFS, NIS 등)들을 공격하게 된다. DNS 스푸핑은 합법적인 네임서버로 위장 (2.c)하여 잘못된 정보를 응답하도록 유도하

는 것이다.

소프트웨어 취약성에 대한 공격은 일반적인 시스템과 네트워크에 알려진 공격으로서 DNS에만 한정된 보안 취약성이라고 할 수는 없다. 하지만, 이를 통해 네임서버에서 악의의 코드를 실행하여 공격자가 의도한 바를 수행(4.a) 하거나, DNS 서버에 대한 버퍼 오버플로우 공격을 통해 시스템의 루트 권한을 획득(4.b)하여 정보를 위·변조 할 수 있으며, 서비스 거부 공격(4.c)을 통해 서비스 자체를 유용하지 않게 할 수도 있다.



[그림 3] DNS 캐쉬 오염(Cache Poisoning)

이 외에도 위에서 언급된 취약점에 대한 공격 형태로 존 트랜스퍼를 통한 위·변조 혹은 관련 정보의 누출, DNS 동적 업데이트(DNS dynamic update)를 통한 위·변조, DNS 존 복제과정에서의 위·변조, 와일드 카드(wildcard) 질의에 대한 응답에서의 무결성 문제, 클라이언트 플러딩(client flooding) 등이 존재하다.

실시간 처리와 이에 대한 신뢰를 대상으로 보안을 요구하는 서비스를 제공하는 웹 사이트의 경우, 제공하는 서비스가 악의적인 목적으로 위·변조(1.a-b-c)되면 해당 시스템이나 사용자는 유, 무형의 손실을 입게 된다. 또한 로컬 사이트에서 프락시(proxy) 시스템을 통해 인터넷을 사용하거나 NSP(Network Service Provider)를 통한 서비스를 제공받는 사용자의 경우에 스터브 리졸버(Stub resolver)의 캐쉬가 오염될 경우 역시 치명적인 보안 문제를 야기하게 된다.

<표 1> DNS 취약성의 분류

분류	취약성	보호 대상
1. 서비스 관점	a. 질의에 대한 널(null) 응답 b. 잘못된 질의 응답 (redirect, inject) c. 서비스거부 (Denial of Service)	데이터/서버 보호
2. 공격형태 관점	a. 패킷가로채기를 통한 위/변조 b. ID추측/질의예측을 통한 위/변조 c. 네임기반공격을 통한 위/변조 d. 신뢰된 서버로의 위장을 통한 위/변조	데이터 보호
3. 공격대상 관점	a. DNS 클라이언트, 네임 서버, 리졸버	데이터/서버 보호
4. 소프트웨어 취약성 관점	a. 네임서버에서의 악성코드실행 b. 네임서버의 루트쉘 권한 획득 c. 서비스 거부 공격	서버 보호
5. 기타	a. 존 트랜스퍼를 통한 위/변조 혹은 정보 누출 b. DNS 다이나믹 업데이트를 통한 위/변조 c. DNS 존 복제과정에서의 위/변조 d. 만능문자 질의에 대한 무결성 e. 클라이언트 플러딩	접근제어/서버/데이터 보호

이와 같이 DNS 시스템의 취약성은 잘못된 데이터 기원 인증과, 무결성의 침해 가능성에서 기인하며 이를 보완할 수 있는 안전한 DNS가 요구된다. 따라서 이에 대한 해결책으로 제시되고 현재까지 지속적으로 연구 개발되고 있는 DNSSEC에 대하여 살펴보도록 한다.[9][10]

3. DNSSEC

지금까지 살펴본 DNS의 취약점을 해결하기 위해 DNS에 대한 보안 확장인 DNSSEC의 개념이 제시되었다.

3.1 DNSSEC의 개요

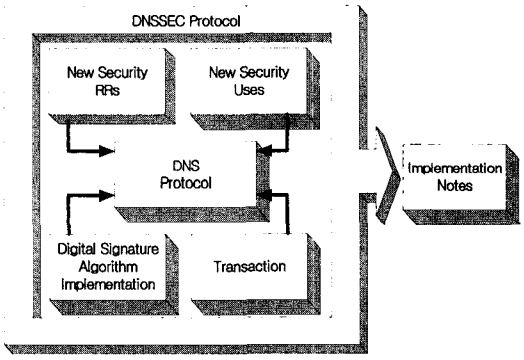
DNSSEC(DNS Security Extensions)은 현재 사용 중인 DNS에서의 보안 문제점을 해결하기 위해 제시된 DNS 확장의 개념이다. 기본적인 개념은 DNS에서 신뢰할 수 없는 DNS 요청에 대한 응답을 디지털 서명하여 악의적인 목적을 갖는 사용자가 데이터를 악용하지 못하게 하는 것으로, 새로운 보안 관련 RR을 바탕으로 디지털 서명 알고리즘을 적용하여 데이터에 대한 인증과 데이터 무결성 서비스를 제공하는 것이며 궁극적으로는 현재의 DNS 트리 구조를 DNSSEC 트리 구조로 바꾸는 데 목적이 있다.[18]

DNSSEC은 명칭에서도 알 수 있듯이 DNS의 보안상의 문제점을 해결하기 위해 기존 DNS 체계에 몇 가지 부분으로 나뉘어진 보안 구조(architecture)를 도입한다. 그러나 이것이 기존에 사용되던 DNS 프로토콜 자체에 대한 큰 변화를 가져오는 것은 아니며 현재의 표준안에서도 DNSSEC의 실제 적용 단계에서의 용이성과 확장성을 위해 DNS 프로토콜 자체에 대한 변화는 최소화시키며 보안기능을 향상시키는 방향으로 정의되어 있다.

3.1.1 보안을 위한 DNS 확장

DNS의 보안을 위해서는 [그림 4]의 DNSSEC 로드맵(roadmap)에서 보듯이 새로운 보안 관련 RR의 도입, 디지털 서명 알고리즘, 다른 프로토콜(프로토콜)과의 연계를 통한 보안 강화 등이 필수적이며 여기에 추가적으로 트랜잭션(transaction)에 대한 보안등이 고려된다.

DNSSEC에서 사용되는 새로운 보안 관련 RR로는 SIG(Signature), KEY, NXT(NeXT) RR이 있다. KEY RR은 존이나 다른 종단 개체의 공개키를 저장하는데 사용된다. 이 공개키는 신뢰사슬(trust chain)의 개념을 따라 상위 존의 개인키로 암호화(encryption)되어 하위 존의 종단 개체에 전송된다. SIG RR은 해당 존의 개인키를 이용하여 생성된 디지털 서명과 그에 관련된 기타 사항에 대해 정의한다. 여기에는 서명에 생성된 암호화 알고리즘과 서명의 만기일 및 서명자 정보 등이 포함된다. NXT RR은 DNS 질의가 도착했을 경우 그 질의에서 요청한 호스트 정보가 존재하지 않을 경우 사용된다. 이는 존재하지 않는 호스트에 대해서도 일일이 NXT RR을 이용하여 응답해야 하므로 과부하(overhead)가 더 커지는 단점이 있기는 하지만 DNSSEC의 보안 체계를 유지한다는 의미에서 꼭 필요한 RR이다.



[그림 4] DNSSEC roadmap

DNSSEC에서는 디지털 서명을 사용하여 전송되는 메시지에 대한 무결성(integrity)를 보장한다. 여기에 사용되는 알고리즘에는 DSA (Digital signature Algorithm), RSA/MD5, Diffie Hellman 알고리즘 등이 있으며 이 중 DSA가 의무적으로 사용되고 있다. DSA는 NIST에서 개발한 전자 서명을 위한 알고리즘으로 이산대수 문제를 기초로 하는데 키 교환이 어렵고 아직 많은 연구가 진행되지 않아 실제 적용 시에 발생 가능한 문제점들이 확인되지 않았다는 단점이 있기는 하지만 서명의 생성이 검증보다도 빠르다는 장점이 있어 DNSSEC에 적합하다. 그 밖의 특징으로는 DNSSEC이 유연(flexibility)하기 때문에 다른 프로토콜이나 응용프로그램과의 연계가 가능하다

는 점 등을 들 수 있다. IPSEC-DNS나 SSH-DNS는 이러한 개념에서 나온 DNSSEC의 보안적인 확장이다.[15]

3.1.2 인증과 무결성

DNSSEC의 가장 큰 문제점은 수신된 요청 메시지로부터 이것이 적법한 절차를 거친 정상적인 메시지인지를 구분하는 것이 불가능하다는 것이다. 따라서 이를 해결하기 위해 수신된 메시지가 적법한 호스트(source host)에서 전송된 것인지에 대한 인증이 필요하다. 더불어 전송중인 메시지가 공격자에 의해 변경되어 결과적으로 심각한 위험을 낳게 되는 경우도 있을 수 있기 때문에 이에 대해 전송중의 무결성 문제도 고려되어야 한다.

1) 기원 인증

: DNS에서는 메시지에 대해 그 소유자가 적법한지를 표시하기 위해 SOA RR을 사용하게 된다. 여기에는 존의 캐쉬 정보 및 이 존에 포함되어 있는 정보의 갱신을 위한 필드들이 정의되어 있는데 이 SOA RR의 정보를 조작하게 될 경우 그 메시지를 수신한 네임 서버는 진위여부를 확인할 수 있는 방법이 없다. 따라서 이러한 위험성을 막기 위하여 데이터의 기원이 된 호스트에 대한 인증 작업이 필수적인데 여기에 사용되는 것이 신뢰사슬 개념이다. 신뢰사슬 개념은 상위 존에서 인정한 네임 서버에 대해서는 해당 존의 개인키를 이용하여 그 네임 서버에서 사용하는 공개키를 암호화하는 방식인데 이와 같은 방식으로 한 등급씩 높은 네임 서버를 거슬러 올라가게 되면 최상위 도메인까지 도달하게 되어 그 네임 서버가 적법하다는 것을 인정받을 수 있게 된다.

2) 메세지 무결성

: DNS에서 또 하나 간과할 수 없는 문제점이 메시지가 전송도중 위조 혹은 변조될 수 있다는 가능성이다. DNS 메시지는 보통 전송에 UDP를 사용하며 그로 인해 효율적이지만, 전송 도중 위험에 노출될 가능성도 높아졌다. 따라서 메시지가 전송 도중에 공격자에 의해 변경이 시도될 경우에 대해 무결성을 보장할 수 있는 기능이 필요하다. DNSSEC에서는 이를 위해 디지털 서명 알고리즘을 사용하는데, 전송할 메시지는 공개되어 있는 알고리즘으로 디지털 서명되어 수신 측에서 생성한 서명 값과의 비교를 통해 무결성에 대한 검사가 수행된다. 만일 무결성 검사에서 위조나 변조 혹은 자연적으로 발생한 메시지 손상에 대해서는 그에 따른 처리를 하기보다는 그 메시지를 제거하고 다시 전송받는 방식을 사용한다.[11][13]

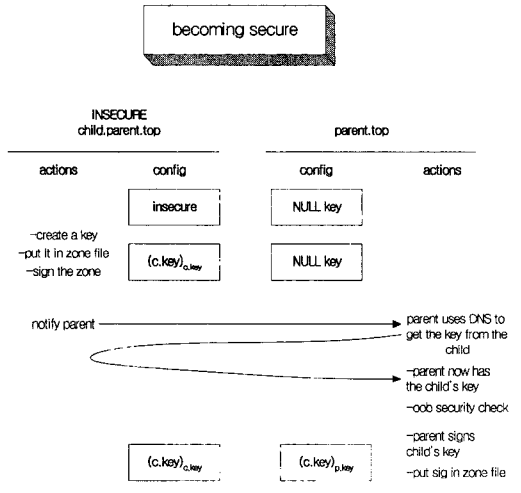
3.2 DNSSEC의 동작 방식

DNSSEC의 동작방식은 크게 신뢰할 수 있는 존에 대해 키를 분배하는 과정과 분배된 키를 이용하여 존의 정보를 서명하는 과정, 그리고 해결 과정에서 확립되는 신뢰 사슬 과정이 있다. [12]

3.2.1 키 교환

임의의 존에 대해 신뢰성을 보장하기 위해서 가장 선행되어야 하는 것이 인증에 사용될 키의 교환을 통한 분배 문제이다. 여기에는 통상적으로 두 가지 방식이 고려되고 있는데 그 첫 번째가 이메일을 통해 키를 요청하는 방식이고 두 번째가 상위의 안전한 도메인으로부터 일련의 과정을 통해 인정받는 방식이다. 이메일을 통해 키를 요청하는 방식은 키에 대한 서명 요청이 들어올 경우 이것이 traceroute 정보와 함께 책임자에게 전송되며 관리 책임자는 요청을 한 측이 보안적

으로 인정할 수 있는 경우 서명된 키를 돌려주는 방식이다. 이 경우는 메일 전송간에 있어서의 보안이 중요하게 다루어지기 때문에 PGP(Pretty Good Privacy)등의 메카니즘을 사용하지만 인증 요구에 대한 처리 비용이 너무 크다는 단점이 있다. 두 번째로 제시된 방식은 [그림 5]에서 보듯이 하위 존에서 공개키 알고리즘을 사용하여 공개키/비밀키 쌍을 생성한 후 생성된 공개키를 자신의 존 파일 내부에 저장해두면 상위 존에서 존 트랜스퍼를 이용하여 하위 존의 공개키를 수신하고 그 공개키를 자신의 개인키로 서명하여 다시 하위 존에 전송하는 방식이다. 이렇게 함으로써 상위 존과 하위 존 사이에는 인정된 키의 분배가 가능해지는데 이 방식은 키의 전송과정 중에 IP rerouting attack에 취약하다는 단점이 있지만 DNS 체계에서 DNSSEC 체계로의 쉬운 전이가 가능하다는 점에서 일반적으로 고려되고 있는 방식이다.



[그림 5] 일반적인 초기 키 교환 방식

3.2.2 존에 대한 서명

DNSSEC에서는 SIG RR을 사용하여 존에 대해 서명된 정보를 저장한다. 이 정보는 어떤 요청에 대해 서버측에서 요청을 한 요소(entity)가 보안적으로 안전한 존인지를 구별할 수 있게 해주고 실제 공격에도 안전성을 제공해줄 수 있다. 존에 대한 서명은 온라인(on- line) 방식과 오프라인(off-line) 방식이 있는데 온라인 방식은 DNSSEC에서 주로 고려되고 있는 방식으로, 상위 존에서 하위 존에 대한 키를 서명하는 방식이며 오프라인 방식은 상위와 하위 관계에 있지 않은, 이미 잘 알려진 보안 지원 존(S-Zone : Secure-aware zone)으로부터 해당 존에 대한 키를 서명받는 방식을 말한다. 이는 보유해야 하는 키의 개수를 한정할 수 있다는 장점이 있다.

3.2.3 신뢰 사슬

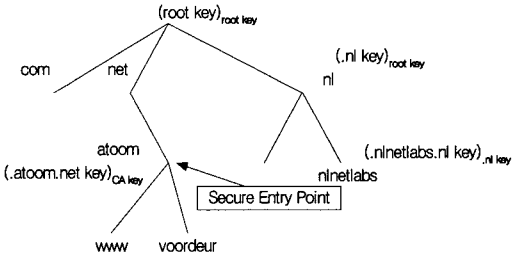
신뢰 사슬은 DNSSEC에서 각 존에 대한 인증에 있어 가장 중요한 개념이라 할 수 있다. DNS는 전세계적으로 엄청난 양의 존을 구성토록 하였다. 이에 대해 보안적인 안전성을 모두 보장하기란 결코 쉽지 않은 일이다. DNSSEC에서는 이를 해결하기 위해 상위 존과 현재 존 사이에 인증을 적용하여 일종의 '신뢰할 수 있는 사슬구조'를 형성하도록 했다. DNS와 DNSSEC이 모두 트리 구조이기 때문에 항상 상위와 하위 존의 관계는 성립이 되므로 각 상위 및 하위 존이 보안적인 연결구조를 갖도록 한 것이다.

모든 존은 공개키 알고리즘을 사용하여 그 존에서 사용할 공개키와 개인키를 소유한다. 개인키는 디지털 서명 알고리즘에서 항상 서명에 사용되고 공개키가 그 서명에 대한 복호화 및 인증에 사용되므로 공개키는 클라이언트 측에 전송되어야한다. 이 공개키는 KEY RR에 저장되며 리졸버가 수신한 메시지는 존 데이터 중에서 응답할 내용과 함께 SIG RR 및 KEY RR을 함께 전송하게 된다. 리졸버는 이 KEY RR의 공개키를 이용하여 SIG RR을 복호화함으로써 인증작업을

수행할 수 있지만 전송 중에 KEY RR의 내용이 공격자로 하여금 변질될 우려가 있어 KEY RR에 대한 신뢰성이 문제가 된다. 따라서 KEY RR의 신뢰성을 보장하기 위해 상위 존의 개인키로 서명하는 것이 바로 신뢰 사슬의 기본 동작이라 할 수 있다. 이렇게 서명된 공개키는 클라이언트에게 전송되어도 그 클라이언트는 그 공개키의 사용이 불가능하다. 이 공개키를 사용하기 위해서는 그보다 상위 존에 있는 공개키가 필요하게 되며 이 공개키 역시 또 그보다 상위 존의 개인키로 서명이 되어 있기 때문에 보다 상위 존의 공개키가 필요하게 된다. 이렇게 연속적으로 거슬러 올라가게 되면 결국 최상위 도메인(TLD : Top Level Domain)에 도달하게 되며 TLD는 자신의 공개키를 서명한 값을 클라이언트 측에 설정해두었으므로 이 최상위 도메인의 키를 이용한 복호화를 통해 단계별로 하위 존의 키를 습득하고, 클라이언트는 결국 자신이 처음에 질의를 보냈던 해당 존의 공개키 값을 복호화할 수 있게 되며 궁극적으로는 원하는 정보를 얻는 것이 가능해진다.

아래의 [그림 6]은 이에 대한 예를 나타낸 것이다. 여기서 $(x)_y$의 개념은 y의 개인키로 x를 암호화한다는 것을 나타낸다. 만일 ".atoom.net"이 ".nlnetlabs.nl"에 대한 DNS 요청을 보냈다면 이 요청에 대한 응답으로 ".nlnetlabs.nl"에서는 $(.nlnetlabs.nl)_{.nlkey}$으로 암호화된 ".nlnetlabs"의 공개키와 함께 ".nlnetlabs"의 개인키로 서명된 메시지를 전송하게 된다. 이 메시지를 복호화 하기 위해서는 ".nlnetlabs"의 공개키를 얻어야 하는데 앞에서 말했듯이 이 키는 $(.nlnetlabs.nl)_{.nlkey}$로 암호화 되어 있으므로 ".nl"의 공개키를 먼저 얻어야 한다. 그러나 ".nl"의 공개키 역시 루트 키(root key)로 암호화되어 있으므로 다시 루트 키를 얻어야 하고 이 루트 키는 사전에 키의 교환 단계에서 .atoom.net에 전달되어 있어야만 한다.

사슬은 중간에 어느 한 고리가 끊어지게 되면 사용이 불가능해진다. 이와 마찬가지로 신뢰 사슬에서도 반복적으로 키를 얻어나가는 과정에서 단 한 개의 사슬이라도 끊어지면, 즉 중간의 한 서버가 신뢰성을 인정받지 못하거나 제대로 동작하지 않는다면 신뢰 사슬의 개념은 파괴되고 이에 대해서는 안전하지 않은 것으로 판단하여 제거하는 과정이 필요하다.



[그림 6] trust chain

4. DNSSEC 동향

4.1 표준화 동향

IETF 내에서 DNS 연구와 발전을 주도하는 워킹그룹은 dnsext (DNS Extensions)와 dnsop (DNS Operations)이다. dnsext는 DNSSEC을 연구하는 각 기관의 실무자들로 구성되어 있으며, DNSSEC을 위한 세부적인 프로토콜을 정의하는 등 실질적인 DNSSEC 기술 전개를 위해 앞장서고 있다. dnsext 워킹그룹이 표준화하고 있는 DNS 보안 기술은 DNSSEC 기술과, 트랜잭션 보안 기술 두 가지로 나뉜다.

현재 DNSSEC 기술은 세부적인 프로토콜 기준이 마련된 상태이며, 구현의 관점에서 복잡성과 효율성에 관한 논의가 이루어지고 있다. 반면 트

랜잭션 보안 기술은 아직까지 표준안 정의를 위한 세부 프로토콜이 논의되고 있는 상황이다.

<표 2> DNSSEC 표준화 동향

IETF Meeting dnsext proceedings			내 용
년도	회차	Issues for DNSSEC	
2000	48th	DNS Security Document Roadmap	향후 DNSSEC 표준화 문서들의 획일성을 유지하기 위한 Roadmap 작성. IPSEC 문서 Roadmap과 유사함.
		GSS-TSIG	TKEY와 GSS-API를 이용한 공유 비밀키 생성
2001	50th	RFC 2535	DNSSEC의 핵심 문서로서 정리 방안
		Child SIG at the parent	• key roll-over를 축소시킴 • DNS 관리자와 하위 존과의 연계를 감소 • 계획되지 않은 key-roll-over를 수행
		DNSSEC Opt-IN	DNSSEC이 가지고 있는 규모의 문제(거대한 존을 어떻게 서명할 것인가)에 대한 논의 등.
		EDNS handling unknown	EDNS0는 알려지지 않은 옵션에 대한 디폴트 핸들링을 명시하지 않는데, 이것은 옵션 공간을 분할해서 서버가 더 나은 행동의 기회를 갖도록 한다.
	51th	Delegation Signer	DS 레코드 최초 제안.
		State of DNSSEC within dnsext	2001년도 당시 DNSSEC의 기술 개발 및 구현 현황과, 개발 역사, 기술 검증 등의 논의.
2002	54th	key-signing-key flag	• key-signing-key와 zone-signing-key의 구분 • flag field → 256: zone-signing-key, 257: key-signing-key
		wildcard optimization	존 내에 wildcard가 포함되어 있지 않음을 증명

<표 2>는 2000년부터 2002년 사이에 IETF 회의에서 논의된 dnsext 워킹그룹의 회의록 내용 중 DNSSEC과 관련된 쟁점 논의 사항을 정리한 것이다. dnsop 워킹그룹은 DNS 서버 운영과, DNS 존 파일 관리에 관한 가이드 라인을 제시하고 관련 문서를 출판하고 있다. DNSSEC과 관련해서 dnsop 워킹그룹은 dnsext와 협력하여 DNSSEC 관리/운영자를 위한 가이드 라인을 제시하고 있다.

4.2 개발 동향

2000년 12월에 개최된 제 49 차 IETF회의에서는 DNSSEC 개발과 관련된 그룹들의 모임이 있었다. 이 모임은 DNSSEC의 실질적인 기술 개발과 구현을 리드하고 있는 워킹그룹과 각국의 연구소가 참여해 '과연 DNSSEC이 인터넷에서 전개 될 수 있는 기술인가?'에 대한 의문에 대해 토론을 시작했다.

현재는 당시의 의문을 시발점으로 실질적인 구현에 초점이 맞추어진 많은 프로젝트가 여러 연구기관에서 진행 중이고, 이의 성과를 토대로 DNSSEC을 지원하는 다양한 툴(tool)이 개발되었으며, 국가 도메인(TLD) 범위로 한정된 제한적인 DNSSEC 테스트베드까지 구축되어 있어, DNSSEC의 전개가 목전에 이르렀다고 할 수 있다.

다음은 각 프로젝트를 진행하고 있는 기관의 최근 활동을 정리한 것이다.

1) NLnet Labs
(http://www.nlnetlabs.nl/dnssec/)
: Ted Lindgreen과 Miek Gieben이 주도하고 있는 DNSSEC 팀의 연구 방향은 키의 안전성을 위한 키 롤오버에 대한 단순한 메커니즘 개발과 국가 최상위 도메인을 보안 진입점(secure entry point)으로 활용하는 것이다. 이미 이에 대한 DNSSEC 테스트 베드가 운영되고 있으며 DNSSEC 지원 리졸버 역시 개발을 마친 상태이다.

2) RIPE-NCC (http://www.ripe.net/disi/)
: RIPE-NCC(Resource IP Europeens

Network Coordinate Centre)는 DNSSEC 관련 프로젝트인 DISI(Deployment Internet Security Infrastructure)를 통해 IETF에서 추진하고 있는 DNSSEC 관련 표준화 작업에 참여하고 DNSSEC 전개를 위한 트레이닝 코스와 워크샵을 개최하는 등 활발한 활동을 보이고 있으며 특히 역 주소 사상(reverse address mapping)을 보호하기 위한 IN-ADDR SEC 프로젝트에 역량을 집중하고 있다.

3) rs.net (http://www.rs.net)

: rs.net이란 스웨덴의 TLD 관리 기관인 SE-NIC의 주도하에 수행중인 Root Server Testbed Network 프로젝트로써 현재 NIC-SE, CR&T, Autonomica, SUNET/KTHNOC, SIDN, NLNetLabs 등이 참여하여 테스트 네트워크를 구성하고 참여를 원하는 TLD에 대해서는 rs.net이 요구하는 사항에 따라, 일정 절차를 통해 네트워크에 가입하여 DNSSEC을 운영할 수 있는 환경을 제공함으로써 보다 활발한 DNSSEC 전개활동을 펴고 있다.

4) NIST (http://snad.ncsl.nist.gov/dnssec/)

: Scott Rose가 주도하고 있는 NIST(National Institute of Standards and Technology)의 DNSSEC 프로젝트에서는 주로 DNSSEC이 기존 DNS 장비 자체의 작업 부하에 얼마만큼의 작업 부하를 더하게 되는지에 대한 성능 정보를 수집하고 있다. 이를 위해 NIST는 자체 테스트베드를 구축하고 쿼리 테스트와 존 트랜스퍼 테스트 등을 통해 DNS 구현 방식에 따르는 성능을 측정하고 있으며, 최종적으로는 보안 확장이 DNS 서버의 성능에 가져올 영향을 측정하는 것을 목표로 하고 있다.

5) NAI Labs

(http://www.nai.com/research/nailabs/network-security/internet-infrastructure.asp)

: Sandra Murphy가 주도하고 있는 인터넷 기반구조 보안(Internet Infrastructure Protection) 프로젝트를 통해서 NAI Lab은 자신들이 보유한 DNS 보안 경험을 바탕으로 IETF의 DNSSEC WG과 함께 DNSSEC 표준화 작업을 수행했으며 지금은 IPv6를 위한 동적 주소 정보와 동적 호스트 정보를 지원하기 위해 초기 DNSSEC을 확장시키는 연구를 진행 중이다.

6) HISL The Johns Hopkins University (http://www.cs.jhu.edu/~smang/sshproject.html)

: Stefan Mangard가 주도하고 있는 HISL (Hopkins Information Security Lab)의 프로젝트는 DNSSEC의 보안 특성을 이용해서 SSH 프로토콜을 개발하고 있다.

5. 결론

지금까지 DNS의 기본 개념 및 취약성과 그와 관련된 해결책 도출을 위한 보안 확장인 DNSSEC에 대해 살펴보았다. DNSSEC을 적용함에 따라 여러 가지 문제점이 발생하게 되는데, 그 첫 번째가 인증에 필요한 여러 정보들로 인한 과부하의 발생이다. 실제로 DNSSEC을 적용할 경우에 한 메시지의 크기가 현재의 6배 정도가 증가하게 된다. 이는 효율성을 위해 UDP를 사용하던 현재의 DNS 시스템에서 대부분 TCP를 사용하도록 만드는 직접적인 계기가 될 것이며 TCP를 사용하게 됨으로써 헤더의 크기 증가와 통신 설정 과정을 거치기 위한 비용 증가를 고려해야 한다. 두 번째가 키 관리의 어려움이다. 사실 DNS는 전세계적으로 사용되고 있는 프로토

콜로써, 구성하고 있는 모든 존에 대해 키를 분배해야 한다면 이는 엄청난 작업이 아닐 수 없다. 게다가 현재 인터넷의 구성 상 전세계를 망라할 수 있는 공개키 기반 시스템이 존재하지도 않을뿐더러 만일 존재한다 하더라도 그 모든 키를 관리하기 위해서 사용되는 트래픽의 증가는 인터넷의 전반적인 성능저하를 감수하지 않을 수 없을 것이다. 마지막으로 관리적인 측면에서도 많은 어려움을 가져오게 될 것이다. DNSSEC 구조를 도입하여 유지 관리하기 위해서는 복잡해진 시스템 구조를 다루기 위한 관리자의 노력이나 관리 정보의 양의 증가에 대한 엄청난 비용이 추가적으로 발생하게 되며 세계적인(global) 키 분배 문제는 아직까지 명확한 해결책이 모색되지 않아 실제로 DNSSEC의 적용은 많은 시간이 필요한 상태이다.

그러나 현재 사용 중인 DNS의 위험성이 점차 커지면서 많은 연구기관에서 DNSSEC의 적용을 위한 노력에 박차를 가하고 있으며 각국에서 계속되는 연구로 새로운 개선안과 보완책들이 제시되고 있다. 이러한 노력의 성과로 국가 도메인에서의 DNSSEC이 구현되는 실적까지도 낳게 되었으며 이러한 실험적이면서도 점진적인 확장은 짧은 시간 내에 DNSSEC이 인터넷 에 적용될 것이라는 가능성을 보여준다.

참고문헌

[1] P. Mockapetris, "DOMAIN NAMES - CONCEPTS AND FACILITIES", RFC 1034, November 1987.

[2] P. Mockapetris, "DOMAIN MAMES - IMPLEMENTATION AND SPECIFICATION", RFC 1035, November 1987.

[3] D. Atkins, R. Austein, "Threat Analysis of The Domain Name System", Internet-Draft, November 2002.

[4] 유신근, 이현우, "DNS 안전운용가이드", CERT/CC-kr, February 2001.

[5] http://www.certcc.or.kr/advisory/ka2002/ka2002-063.txt, "DNS Resolver 라이브러리의 버퍼오버플로우 취약점", CERT/CC-kr, June 2002.

[6] Steven M. Bellovin, "Using the Domain Name System for System Break-ins",.

[7] Paul Vixie, "DNS and BIND security Issues", Fifth USENIX UNIX Security Symposium in Salt Lake City. Utah, June 1995.

[8] 조용상, "DNS/BIND 관련 최근 취약점과 그 대책 분석 보고서", CERT/CC-kr, January 2000.

[9] Davidowicz,"Domain Name System (DNS) Security", Network Magazine, January 2000.

[10] Lioy, Maino, Marian, Mazzocchi, "DNS Security",Terena Networking Conference, May 22-25,2000.

[11] Diane Davidowicz, "Domain Name System(DNS) Security".

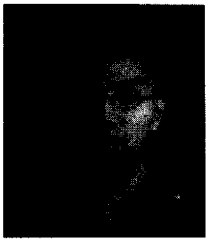
[12] R. Gieben, "Chain of Trust The Parent-child and keyholder-keysigner relations and their communication in DNSSEC", Stichting NLnet Labs, November 13,2000.

[13] Christoph L. Schuba, Eugene H. Spafford, "Countering Abuse of Name-based Authentication", COAST

LAB Department of Computer Sciences Purdue University West Lafayette, 1994.
[14] R. Arends, M. Larson, D. Massey, S. Rose, "Resource Records for the DNS Security Extentions", Internet-Draft, October 2002.
[15] R. Arends, R. Austein, M. Larson, D. Massey, S. Rose, "DNS Security Introduction and Require- ments", Internet-Draft, February 2003.
[16] R. Arends, M. Larson, R. Austein, D. Massey, S. Rose, "프로토콜 Modifications for the DNS Security Extensions", Internet-Draft, March 2003.
[17] D. Conrad, "Indicating Resolver Support of DNSSEC", RFC 3225, December 2001.
[18] D. Eastlake, "Domain Name System Security Extension", RFC 2535, March 1999.
[19] Ross Wm. Rader, "One History of DNS", www.byte.org/one-history-of-dns.pdf , April, 2001

김 학 주
2003년 2월 성균관대학교 전기전자컴퓨터공학부 졸업(학사),
2003년 3월 - 성균관대학교 대학원 컴퓨터 공학과(석사과정)

<관심분야> 네트워크 보안, 시스템 보안, 분산 컴퓨팅



윤 민 우
2000년 2월 성균관대학교 전기전자컴퓨터공학부 졸업(학사),
2002년 11월 ~ 성균관대학교 인터넷 관리 기술 연구실

<관심분야> 네트워크 관리(Active, GRID), IP Mobility, IP Multicast, IPv6

임 형 진
1998년 2월 한림대학교 컴퓨터공학과 졸업(학사),
2001년 8월 성균관대학교 정보통신대학원 졸업(석사),
2003년 3월 ~ 성균관대학교 대학원 컴퓨터공학과(박사과정)

<관심분야> 네트워크 관리, 네트워크 보안, 시스템 보안

송 관 호
1980년 서울대학교 공과대학 전자공학과(학사),
1984년 한양대학교 대학원 전자공학과(석사),
1995년 광운대학교 대학원 자통신공학과(박사),
1997년 서울대학교 행정대학원 정보통신정책과 수료,
1987년 ~ 1995년 한국전산원 초고속국가망구축실장(연구위원),
1995년 ~ 1997년 숭실대학교 정보과학대학원 겸임교수,
1996년 ~ 1997년 한국전산원 표준본부 본부장,
1998년 ~ 1999년 Visiting Professor University of Maryland,
1999년 한국전산원 국가정보화센터 단장,

광운대학교 전산대학원 겸임교수,
1999년 ~ 한국인터넷정보센터 원장,
2002년 ~ 건국대학교 정보통신대학 겸임교수

<관심분야> 인터넷응용, 초고속인터넷

정 태 명

1981년 2월 연세대학교 전기공학과 졸업(학사),
1984년 6월 일리노이 주립대학 전자계산학과 졸업(학사),
1987년 12월 일리노이 주립대학 컴퓨터공학과 졸업(석사),
1995년 8월 퍼듀 대학 컴퓨터공학 (박사),
1984년 ~ 1987년 Waldner and Co., System Engineer,
1987년 ~ 1990년 Bolt Bernek and Newman Labs. Staff Scientist
1995년 9월 - 성균관대학교 정보통신공학부 부교수

<관심분야> 실시간시스템, 네트워크 관리, 네트워크 보안, 시스템 보안, GRID 네트워크, 전자상거래