Port Forwarding에 기반한 네트워크 통신보호 시스템의 설계 및 구현[1]

권 문 상 °, 이 성 우, 조 유 근
서울대학교 컴퓨터공학과

Design and Implementation of Secure Network Communication System based on 'Port Forwarding' Mechanism

Moon-Sang Kwon, Sung-Woo Lee, and YooKun Cho
Department of Computer Engineering, Seoul National University

요 약

인터넷이 월드 와이드 웹(World Wide Web)의 인기에 힘입어 일상 생활의 일부가 되고는 있지만 보안 요건 중의 하나인 '통신의 비밀성(Confidentiality) 요건'을 만족시키지 못하고 있기 때문에 많은 네트워크 응용에서 문제가 되고 있다. 새로운 네트워크 프로토콜인 IPv6[4]에서는 프로토콜 단계에서의 암호화 서비스 제공을 통해 네트워크 통신을 보호해 줄 것으로 기대되지만 아직 표준으로 확정되지 않았으며 실제 네트워크 통신에 언제부터 사용 가능할지 예측할 수도 없다. 이러한 상황에서, 기존의 네트워크 응용들에 대해 암호화 서비스를 제공하는 방법은 원천코드(source code)를 변경하여 다시 컴파일 하는 수밖에 없다.

본 논문에서는 '포트 포워딩(port forwarding)' 기법을 사용하여 클라이언트/서버 모델로 동작하는 기존 TCP/IP 네트워크 응용들의 원천코드(source code) 변경 없이 네트워크 통신을 보호할 수 있는 *네트워크 통신 보호 시스템*을 설계하고 구현하였다

1. 서 론

군사적, 학술적 목적으로 연구 개발되었던 인터넷은 월드 와이드 웹(World Wide Web)의 인기에 힘입어 원래의 학술적, 군사적 목적뿐만 아니라 상업적, 업무적 용도로 우리들의 일상 생활에 깊숙이 파고들고 있다. 그러나, 이러한 인터넷의 일반화에도 불구하고 아직까지 보안 요건 중의 하나인 '비밀성(Confidentiality)' 요건을 만족시키지 못하고 있어 문제가 되고 있다. 인터넷에서의 통신은 대부분의 경우 그대로 노출되고 있다. 이러한 '인터넷에서의 통신내용 노출 문제'는 일찍부터 지적되어 왔으나 아직까지 완벽한 해결책이 제시되지 못하고 있다

인터넷 통신을 보호하기 위해 사용되고 있는 일반적인 방법은 각 네트워크 응용마다 독자적인 암호화 알고리즘 및 인증 프로토콜을 구현하여 사용하는 것이다. 이 방법은 사용되는 네트워크 프로토콜이나 하부 시스템의 종류에 관계없이 항상 통신내용을 보호할 수 있다는 점에서 유용한 방법이나, 네트워크 응용을 만들 때마다 암호화 알고리즘을 구현해야 하는 것이 문제이다. 또한, 기존의 네트워크 응용프로그램들에 대해서 원천코드를 변경한 후 다시 컴파일 해야만 하는 문제가 있다.

동일한 코드를 반복적으로 만들어야 하는 문제는 암호화 라이브러리를 사용함으로써 어느 정도 극복될 수 있으며, 대표적인 것이 SSL(Secure Socket Library)이다[6]. 프로그래머는 단순히 암호화 라이브러리에 있는 API(Application Programming Interface)를 사용하고 링크함으로써 자신이 지원하고자 하는 암호화서비스를 간단히 구현할 수 있다. 그러나, 암호화 라이브러리를 사용하더라도 기존 네트워크 응용프로그램들에 대해서 암호화 서비스를 제공하려면 원천코드를 변경하지 않을 수 없다.

네트워크 통신을 보호하는 또 다른 방법 중의 하나는 프로토콜 단계에서 통신보호 서비스를 제공하는 것이다. 프로토콜 단계에서 네트워크 데이터에 대한 보호 기능을 제공하는 경우 기존 네트워크 응용들이나 새로 만들어질 네트워크 응용에서는 데이터 보호에 대해 전혀 신경 쓸 필요가 없기 때문에 가장 바람직한 방법이라고 할 수 있다.

현재 사용되고 있는 IPv4 프로토콜을 대체할 새로운 네트워크 프로토콜인 IPv6에서는 프로토콜 단계에서의 암호화 서비스를 제공할 것으로 예상된다[4]. 그러나, 아직까지 표준이 완성되지 않았으며, 이 프로토콜이 구현되어 실제 네트워크 통신에 사용되기까지는 적지 않은 시간이 소요될 것으로 예상된다.

본 논문에서는 *포트 포워딩(port forwarding) 기법*을 사용하여 *클라이언트/서버 방식*으로 동작하는 기존의 TCP/IP 네트워크 응용들

[1]본 연구는 과학기술부의 특정연구개발사업인 "Internet을 위한 방화벽 및 네트워크 통신 보호 시스템 개발(과제번호 sw-09-01)"의 지원에 의해 이루어졌다.

에 대해 원천코드의 변경 없이 암호화 서비스를 제공하여 네트워크 통신 내용을 보호해 주는 *네트워크 통신보호 시스템*의 설계 및 구현에 대해 설명하고 있다.

네트워크 통신보호 시스템을 설계하고 구현하는데 있어서 우리가 고려한 사항은 다음과 같다.

- 기존 TCP/IP 응용프로그램들에 대해 암호화 서비스를 제공한다.
- 기존 응용프로그램들의 원천코드에 대한 변경은 필요 없거나 최소한이 되도록 한다
- 사용자는 암호화 서비스의 존재를 알 필요가 없도록 한다.
- 중복적인 코딩이 필요 없도록 한다.

본 논문의 구성은 다음과 같다. 2장에서는 네트워크 통신보호 기법으로 사용된 *포트 포워딩(port forwarding)*에 대해 설명하며, 3장에서는 네트워크 통신보호 시스템의 전체 구조에 대해, 4장에서는 통신보호 시스템의 구현 및 사용 결과에 대해 설명하며, 마지막으로 5장에서 결론 및 향후 과제를 제시하고 있다.

2. 포트 포워딩(port forwarding)

인터넷 통신의 기본이 되는 클라이언트/서버 방식의 TCP/IP 통신은 다음과 같은 네 개의 값으로 유일하게 결정된다.

(클라이언트의 주소, 포트, 서버의 주소, 포트)

이를 그림으로 표현하면 그림 1과 같다

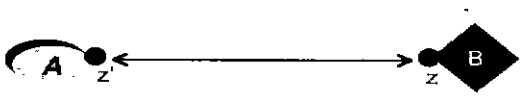

그림 1. 클라이언트/서버 통신

그림 1은 클라이언트 A가 서버 B의 z번 포트로 네트워크 연결을 수행한 결과를 나타내고 있다. 이러한 인터넷 통신은 그림 2와 같이 변형된 형태로 수행할 수 있다.

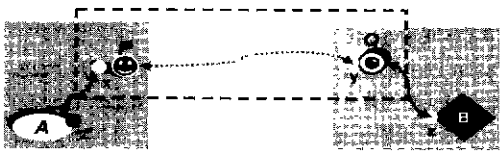

그림 2. 포트 포워딩을 사용한 클라이언트/서버 통신

그림 2에서 P는 클라이언트 A와 같은 시스템에서 실행되고 있는 프로세스를, Q는 서버 B와 같은 시스템에서 실행되고 있는 프로세스를 나타낸다. 프로세스 P는 x번 포트를 개방하고 있으며, 프로세스 Q는 y번 포트를 개방하고 있다. 클라이언트 A는 서버 B의 z번 포트로 연결하는 대신 같은 시스템에서 실행되고 있는 프로세스 P에 x번 포트를 통해 네트워크 연결을 만든다. 네트워크 연결 요청을 받은 P는 프로세스 Q와 y번 포트를 통해 네트워크 연결을 만들고, Q는 다시 서버 B와 z번 포트를 통해 네트워크 연결을 만든다. 이 후의 통신은 P와 Q를 경유하여 이루어지게 된다. 이와 같이 동작하는 네트워크 통신 방식을 *'포트 포워딩(port forwarding)'*이라고 부른다.

포트 포워딩 방식의 네트워크 통신은 일반적인 네트워크 통신에 비해 약 3배의 비용이 더 들게 된다. 즉, A와 P, P와 Q, Q와 B 사이의 연결등 세 개의 클라이언트/서버 연결이 만들어진다. 하지만, 이 방식을 적절히 이용하면 클라이언트/서버 방식으로 동작하는 기존의 TCP/IP 네트워크 응용프로그램들에 대해 암호화 서비스를 쉽게 제공할 수 있다.

즉, 그림 2에서 프로세스 P는 클라이언트 A가 보내온 데이터를 암호화한 후 Q에게 보내고, Q는 P로부터 수신한 데이터를 해독한 후 B에게 보내면, P와 Q 사이의 네트워크 통신 내용을 보호할 수 있다 A와 P, Q와 B 사이의 통신은 같은 시스템 안에서 이루어지는 통신이기 때문에, 운영체제가 메모리에 대한 보호 서비스를 제공하는 경우 안전하다라고 가정할 수 있다. 메모리에 대한 보호 서비스는 C2 레벨 이상의 보안 등급을 지원하는 운영체제라면 기본적으로 제공되는 것이다[10].

포트포워딩에 기반한 네트워크 통신보호 기법은 [5]에서 처음 제시하고 있으나, 이는 포트포워딩 전용시스템이 아니며 중앙 집중적인 키관리 및 인증 프로토콜을 제공하지 않고 있다.

3. 네트워크 통신보호 시스템 구조

구현된 네트워크 통신보호 시스템은 그림 2에서 P의 역할을 수행하는 *지역 통신보호 대리자(Local Secure Communication Agent,* 이하 *LSCA)*와 Q의 역할을 수행하는 *원격 통신보호 대리자(Remote Secure Communication Agent,* 이하 *RSCA)*, 키관리 및 인증 프로토콜을 지원하는 *키서버(Key Distribution Center,* 이하 *KDC)*, 그리고 LSCA와 RSCA를 각각 관리하는 *지역 통신보호 프로그램 관리자*와 *원격 통신보호 프로그램 관리자* 등 다섯 개의 프로그램들로 구성되어 있다 이들 프로그램들의 관계를 그림으로 표현하면 그림 3과 같다.

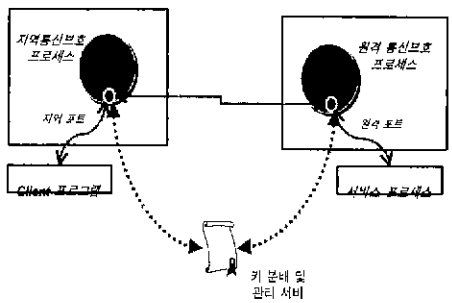

그림 3. 통신보호 시스템 구성

클라이언트 프로그램은 LSCA 프로세스에 지역포트를 통해 연결하고, LSCA 프로세스는 RSCA 프로세스의 원격포트로 연결한다. RSCA 프로세스는 서버프로세스에 연결하기 전에 LSCA 프로세스와 인증 프로토콜 수행을 통해 상호인증 및 세션키 공유를 끝낸다. 인증이 성공적으로 끝나면 두 통신보호 프로세스 사이의 통신은 데이터 암호화를 통해 보호되어 수행된다. 인증이 끝나면, LSCA 프로세

스는 자신이 원하는 서버 프로세스에 대한 정보를 RSCA에게 전달하고, RSCA는 최종 목적지에 일반 네트워크 연결을 만듦으로써 클라이언트에서 서버 프로세스까지의 최종 네트워크 경로를 만들게 된다.

LSCA 프로그램에는 여러 개의 통신보호 프로세스가 있을 수 있으며, 이들 프로세스들은 지역 포트 번호에 의해 구분된다. 하나의 지역 포트에는 *RSCA의 주소 및 포트 정보*와 *서비스 프로세스*에 대한 정보가 지정되어 있으며, 관리자는 LSCA 프로그램 관리자 프로그램을 이용하여 LSCA 프로그램에 연결한 후 새로운 통신보호 프로세스를 생성하거나 기존의 통신보호 프로세스를 제거할 수 있다.

RSCA 프로그램의 경우에도 여러 개의 통신보호 프로세스들이 존재할 수 있으며, 관리 프로그램을 통해 연결한 후 현재 연결되어 있는 네트워크 연결들을 관리할 수 있다.

4. 네트워크 통신보호 시스템 구현

통신보호 시스템에서 사용하고 있는 암호화 알고리즘은 Blowfish, DES, Triple-DES, IDEA, TMSPEED([1], [2], [8], [9]) 등이며, 인증 프로토콜은 KryptoKnight, RSA 기반, *"새로운 인증 프로토콜"*([7], [3]) 등이다. "새로운 인증 프로토콜"은 네트워크 통신보호 시스템을 구현하면서 새롭게 만든 인증 프로토콜이다

프로그램은 *Solaris 2.5.1*에서 구현되었으며, 테스트는 *SUN UltraSPARC 170MHz*를 사용하는 *일반 이더넷 환경*에서 수행하였다 표 1은 LSCA와 RSCA 프로그램을 같은 시스템에서 운영한 결과(실험 A)와, 인접한 서로 다른 두 시스템에서 운영한 결과(실험 B)이다. 인증 프로토콜은 KryptoKnight[7]을 사용하였다. 전송 속도는 8 MByte의 데이터를 클라이언트 프로그램에서 서버 프로그램으로 전송하는데 걸린 시간을 측정하여 구하였다.

암호화 알고리즘	실험 A (Mbits/sec)	실험 B (Mbits/sec)
3-DES	2.67	1.68
Blowfish	8.00	4.00
DES	5.82	3.37
IDEA	2.91	2.06
TMSPEED	0.10	0.10
일반	64.00	7.11

표 1. 통신보호 시스템 전송 속도 측정 결과

*"일반"*은 네트워크 통신보호 시스템을 경유하지 않고 클라이언트에서 직접 서버로 네트워크 데이터를 전송하는 경우이다 *TMSPEED* 암호화 알고리즘의 경우 암호화 알고리즘 수행에 소요되는 시간이 워낙 크기 때문에 네트워크를 통한 시간 지연이 전체 소요시간에 거의 영향을 주고 있지 못함을 알 수 있다. 이는 *TMSPEED* 알고리즘이 속도보다는 안전성에 기반을 두고 설계 및 구현되었기 때문이다[9].

속도면에서 볼 때, 암호화 서비스는 *telnet*같이 인터랙티브한 프로그램에 대해 암호화 서비스를 제공하는데 적합할 것으로 판단되며, 이러한 목적으로는 *TMSPEED* 알고리즘도 사용 가능할 것으로 생각된다. 구현된 네트워크 통신보호 시스템은 *telnet*이나 *pop3* 등에 성공적으로 암호화 서비스를 제공할 수 있었다.

5. 결론 및 향후 과제

본 논문에서는 기존의 TCP/IP 네트워크 응용들에 대해 원천코드 변경 없이 네트워크 통신내용을 보호할 수 있는 *네트워크 통신보호 시스템*을 설계하고 구현하였다. 네트워크 통신보호는 *'포트 포워딩 (port forwarding) 기법'*에 기반하고 있으며, *'클라이언트/서버 방식으로 동작하는 일반적인 네트워크 응용들'*에 대해 암호화 서비스를 제공할 수 있다. 단, 클라이언트 프로그램은 지역 포트를 명령라인에서 지정할 수 있어야 한다.

현재의 네트워크 통신 보호 시스템은 클라이언트 프로그램을 실행할 때 명령라인에서 지역 포트를 지정할 수 있어야 하지만 이를 자동적으로 수행할 수 있게 함으로써 사용자가 전혀 통신보호 시스템의 존재를 인식할 수 없도록 해야 할 것이다. 이는 쉘 스크립트로 기존 네트워크 클라이언트 프로그램을 감싸거나, 동적으로 링크 되는 네트워크 라이브러리를 변경함으로써 구현할 수 있을 것이다.

[참고 문헌]

[1] B. Schneier, "Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish)", Fast Software Encryption, Cambridge Security Workshop Proceedings (December 1993), Springer-Verlag, 1994, pp. 191-204.

[2] National Bureau of Standards, Data Encryption Standard, U.S. Department of Commerce, FIPS Publication 46, Jan 1977.

[3] Rivest, R. L, Shamir, A. and Adle-man, L. M., "A method for obtaining digital signatures and publickey cryptosystems", CACM, vol. 21, 1978, pp. 120-126.

[4] draft-ietf, Stephen Kent, Randall Atkinson, "Security Architecture for the Internet Protocol"

[5] Tatu Ylonen, "SSH -- Secure Login Connections over the Internet", Proc. of the Sixth USENIX Security Symposium July 22-25, 1996

[6] Alan O. Freier, Philip Karlton, Paul C. Kocher, "The SSL Protocol Version 3.0"

[7] Bird, R., Gopal, I, Herzberg, A., Janson, P., Kutten, S., Molva, R., and Yung, M, "The KryptoKnight Family of Light-Weight Protocols for Authentication and Key Distribution", IEEE/ACM Transactions on Networking 3(1), pp.31-41, 1995.

[8] X. Lai, J. Massey, and S Murphy, "Markov Ciphers and Differential Cryptanalysis," Advances in Cryptology--EUROCRYPT '91 Proc., Springer-Verlag, 1991, pp. 17-38.

[9] 이문규, 박근수, 조유근, "함수 풀 및 삼중 암호화에 기반한 개선된 SPEED 암호시스템", 정보과학회 1998년 가을

[10] Donald C. Latham, "Orange Book: Trusted Computer System Evaluation Criteria & Security Evaluation Criteria for Firewall Systems". Department of Defense, December 1985