범용 운영체제를 위한 계층적 패치 분배 구조에서의 효과적인 시스템 인증 방법

배성재 정만현 조재익 문종섭
고려대학교 정보보호기술연구센터
{baeseongjae, manhyun4, chojaeik, jsmoon}@korea.ac.kr

Effective Authentication Method for Hierarchical Patch Distribution

Seongjae Bae Manhyun Jung Jaeik Cho Jongsub Moon
Korea University Graduate School of Information Management & Security

요 약

네트워크가 광범위 하게 발달함에 따라 악의적인 공격도 늘어나고 있다. 악의적인 공격을 막는 방법으로 침입을 차단하는 프로그램을 설치하는 것 외에도 시스템을 패치하는 것은 중요한 부분을 차지한다. 패치 분배 시스템은 대규모 네트워크 상의 시스템에 패치를 분배할 때 부하 분산을 하기 위하여 계층적 구조로 설계된다. 본 논문에서는 계층적 패치 분배 구조에서 상위 계층이 하위 계층을 효과적으로 인증하기 위한 방법을 제안한다.

1. 서론

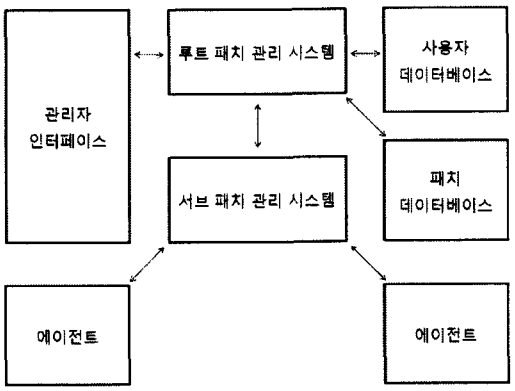
시스템의 취약점을 이용한 공격이 늘어남에 따라 이에 대한 대비책으로 관리자는 시스템을 패치 한다. 네트워크 상에서 다양한 플랫폼을 가진 시스템의 취약점을 패치 하기 위해서는 체계적인 패치 분배 시스템이 필요하다. 대규모의 네트워크 상에서 시스템으로 패치 분배 시에 패치를 분배하는 중앙 서버의 부하를 줄여 효율성을 증대 하기 위해 계층적 패치 분배 구조를 고려해야 한다[1]. 또한 계층적 패치 분배 구조에 적합한 인증을 통한 안전한 패치 분배에 대하여 고려해야 한다

본 논문에서는 계층적 패치 분배 구조에 대하여 알아보고 계층적 패치 분배 구조에 적합한 인증 구조에 대하여 제안한다.

2. 계층적 패치 분배 구조

대규모 네트워크 환경에서 패치를 효율적으로 분배하기 위해서는 패치 분배 시스템을 다중 서버로 구성한다. 그래서 한 개의 루트 패치 관리 시스템을 두고 하위에 여러 개의 서브 패치 관리 시스템을 구축한다. 패치 파일은 서브 패치 관리 시스템을 통해서만 에이전트로 분배 된다. 관리자는 WEB user interface 로 루트 패치 관리 시스템을 통제한다. 루트 패치 관리 시스템의 기능으로는 가장 중요한 패치 파일 분배 기능이 있고, 에이전트 관리, 서브 패치 관리 시스템 관리, 패치 현황 분석, 패치 파일 분배 예약 기능, 패치 정책 관리 등이 있다. 또한 에이전트로 패치 파일을 분배할 때 그룹화 정책을 적용하여 그룹별로 일괄적으로 관리한다[2].

루트 패치 관리 시스템에서만 DB 와 패치 파일을 관리하고 서브 패치 관리 시스템에서는 루트 패치 관리 시스템으로부터 패치 파일을 받아서 에이전트로 분배하는 기능만 수행한다. 아래 그림은 이러한 사항을 고려한 패치 분배 시스템 구조를 나타낸다.



[그림 1] 패치 분배 시스템 구조

위와 같은 계층 구조에는 보안적인 측면을 고려하여 몇 가지 제약 조건이 있다. 첫 번째로 안전하게 패치 파일을 분배하기 위하여 인증은 루트 패치 관리

* 본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음.
(IITA-2006-(C1090-0603-0025))

시스템을 통해서만 이루어 진다. Kerberos 프로토콜의 Key Distribution Center (KDC) 인증 구조와 같이 인증이 한 개의 서버 중심으로만 이루어지므로 여러 서버 간에 키에 대한 동기화 작업이 필요 없어서 키 관리의 편의성을 가진다[3]. 두 번째로 에이전트는 루트 패치 관리 시스템과 서브 패치 관리 시스템의 IP 주소를 저장하지 않는다. 또한 서브 패치 관리 시스템도 루트 패치 관리 시스템의 IP 주소를 저장하지 않는다. 그래서 처음으로 연결을 시도할 때 하위 계층에서 상위 계층으로 연결 요청을 하지 못한다. 그 이유는 악의적인 공격자가 에이전트 또는 서브 패치 관리 시스템으로 위장하여 상위 시스템으로 접근 하는 것을 막기 위함이다. 결과적으로 상위 계층에서 하위 계층으로만 인증 요청을 한다. 위와 같은 계층적 구조를 가진 패치 분배 구조에서는 두 가지 제약 사항을 고려한 새로운 인증 구조가 필요하다.

3. 관련 연구

3.1 인증 구조

인증 구조에서 필수 조건은 상호 간의 신원을 확인하고 세션키를 교환하는 것이다. 세션키는 A 와 B 가 서로 통신을 하기 원할 때 일회적으로 사용하는 키 값이다. 세션키를 교환하는 방법에는 크게 두 가지 방법이 있다. 첫 번째로는 대칭키 교환 방법이 있고 두 번째로는 공개키 교환 방법이 있다. 키 교환 방법에 있어서 가장 큰 문제점은 기밀성과 적시성을 확보하는 것이다. 기밀성을 확보하지 못하면 키를 교환할 때 키 값이 노출되게 된다. 그러면 악의적인 공격이 가능하게 된다. 그래서 기밀성을 확보하기 위하여 전송하는 메시지를 암호화한다. 반면에 적시성을 확보하지 못하면 리플레이 공격에 취약하다. 리플레이 공격이란 악의적인 공격자가 암호화 되거나 그렇지 않은 메시지를 중간에서 도청하여 가지고 있다가 나중에 필요할 때 그 메시지를 그대로 보내서 그에 대한 동작을 하게끔 조작하는 공격을 말한다. 적시성을 확보하기 위해서는 두 가지 방법이 있다. 첫 째는 타임스탬프를 기록하는 방법이고 둘 째는 Challenge 그리고 Response 방식을 사용하는 것이다. 타임스탬프를 기록하는 방식은 A 라는 사용자가 B 라는 사용자에게 메시지를 보낼 때 그 시간을 기록한다. B 라는 사용자가 A 가 기록한 타임스탬프 값과 B 가 메시지를 전달 받은 시간의 차이 값을 메시지 전달 속도와 비교하여서 중간에 변조된 것을 확인한다. 이 방식을 이용하면 메시지가 중간에 변조된 것을 확인할 수 있지만 A 와 B 사이에 반드시 시간의 동기화가 이루어져야 한다는 한계점을 가진다.

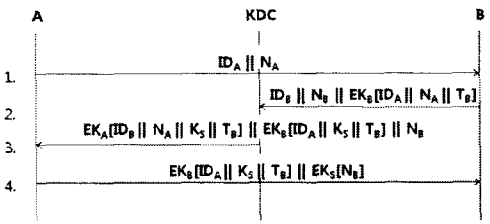
Challenge 그리고 Response 방식은 A 라는 사용자가 B 에게 임의의 값을 전달(Challenge) 하면 B 라는 사용자는 A 로부터 전달(Response) 받은 임의의 값을 A 로 전달하게 된다. 여기서의 문제점은 HTTP 같은 비연결지향 전송에는 적합하지 않다. 왜냐하면 매번 메시지를 전달하기 전에 Challenge 그리고 Response 절차를 수행하기 때문에 오버헤드가 발생하게 된다. 따라서 HTTP 같은 프로토콜에서는 타임스탬프를 이용하는 방식이 적합하다[4].

3.2 대칭키 기반의 인증 구조

신원 확인과 세션키를 교환하는 방법 중에 첫 번째 방법은 아래와 같은 방식인 대칭키를 이용한 방법이다.

<표 1> 대칭키 방식 표기법

A	A 라는 정당한 실체
B	B 라는 정당한 실체
KDC	키 분배 센터
ID_A	A 실체의 신원
ID_B	B 실체의 신원
N_A	A 가 생성한 임의의 값
N_B	B 가 생성한 임의의 값
K_A	A 의 대칭키
K_B	B 의 대칭키
K_S	A 와 B 의 세션키
E	암호화
T_B	B 의 타임스탬프 값



[그림 2] 대칭키 기반의 인증 절차

A 와 B 사이에서 사용할 K_S 를 교환하기 위해서 1 번 과정에서 A 는 B 에게 ID_A 와 N_A 를 평문으로 전송한다. 나중에 N_A 는 K_S 와 함께 암호화 되어 다시 A 로 돌아와서 A 에게 적시성을 보증해준다. 2 번 과정에서 B 가 KDC 에게 K_S 가 필요함을 알린다. 이러한 메시지는 ID_B 와 N_B 를 포함한다. 추후에 N_B 는 K_S 와 암호화 되어 다시 B 로 돌아와서 B 에게 적시성을 보증해준다. KDC 에 보내는 B 의 메시지는 또한 K_B 로 암호화된 블럭을 포함한다. 이 때 K_B 는 KDC 와 B 사이에 쓰이는 대칭키이다. 암호화된 블럭에는 A 에 대한 정보가 담겨 있기 때문에 KDC 가 A 에게 메시지를 전송한다. 또한 이 블럭에는 인증에 필요한 제안된 만료시간과 N_A 를 포함한다. 3 번 과정에서 KDC 는 A 에게 N_B 를 전송하고 또한 K_B 로 암호화된

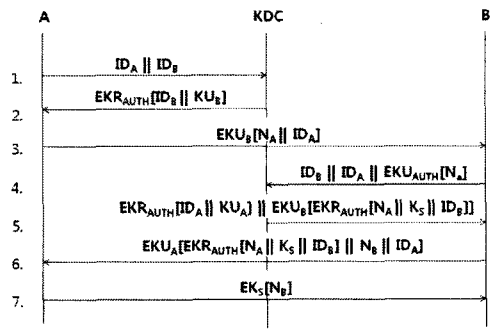
블럭을 전송한다. 이 블럭은 추후의 인증을 위한 티켓으로 사용된다. 그리고 KDC 는 A 에게 K_A 로 암호화된 블럭을 전송한다. 이 블럭을 통해서 A 는 다음과 같은 사항을 확인할 수 있다. ID_B 로는 B 가 A 로부터 처음 전송된 메시지를 잘 받았음을 확인할 수 있고, N_A 로는 전송된 메시지가 리플레이 공격을 받지 않은 적절한 메시지임을 확인할 수 있다. 이 블럭은 또한 K_S 와 T_B 를 제공한다. 4 번 과정에서는 A 에서 B 로 티켓을 전송하고 K_S 로 암호화한 N_B 를 같이 전송한다. 티켓은 K_S 를 제공하여서 암호화된 N_B 를 복호화할 수 있다. B 는 N_B 를 통하여 중간에서 리플레이 공격을 받지 않고 A 로부터 온 정상적인 메시지임을 확인한다[5].

3.3 공개키 기반의 인증 구조

신원 확인과 세션키를 교환하는 방법 중에 두 번째 방법은 아래와 같은 방식인 공개키를 이용한 방법이다.

<표 2> 공개키 방식 표기법

A	A 라는 정당한 실체
B	B 라는 정당한 실체
KDC	키 분배 센터
ID_A	A 실체의 신원
ID_B	B 실체의 신원
N_A	A 가 생성한 임의의 값
N_B	B 가 생성한 임의의 값
KU_{AUTH}	KDC 의 공개키
KR_{AUTH}	KDC 의 개인키
KU_A	A 의 공개키
KR_A	A 의 개인키
KU_B	B 의 공개키
KR_B	B 의 개인키
K_S	A 와 B 의 세션키
E	암호화



[그림 3] 공개키 기반의 인증 절차

1 번 과정에서는 A 가 KDC 에게 B 와 보안 연결할 것을 알리면 2 번 과정에서 KDC 는 A 에게 KU_B 를 전달한다. 3 번 과정에서는 A 는 B 에게 보안 연결할 것을 알리고 KU_B 로 N_A 와 ID_A 를 암호화해서 전송한다. 4 번 과정에서 B 는 KDC 에게 KU_A 와 K_S 를 요청한다. B 는 N_A 를 KU_{AUTH} 로 암호화해서 전송하는데 이는 KDC 로부터 안전하게 N_A 를 확인하게 한 후 N_A 에 맞는 K_S 를 제공받기 원하기 때문이다. 5 번 과정에서는 KDC 가 B 에게 KU_A 와 $\{N_A, K_S, ID_B\}$ 정보를 돌려준다. K_S 와 N_A 가 같이 전송됨으로써 K_S 가 새로 생성된 것임을 나중에 A 로 전송되어 A 에게 적시성을 보증한다. 3 가지 항목은 KR_{AUTH} 로 암호화 되는데 그 이유는 전송된 데이터가 KDC 가 보낸 것을 B 에서 확인하기 위한 것이다. 또한 KR_{AUTH} 로 암호화 된 것을 다시 KU_B 로 암호화한 것은 B 를 제외한 다른 사용자가 3 가지 항목을 이용하여 A 로 접근하는 것을 막기 위함이다. 6 번 과정은 B 에서 A 로 KR_{AUTH} 로 암호화된 $\{N_A, K_S, ID_B\}$ 를 전송한다. 또한 N_B 와 ID_A 를 전송한다. 전송하는 모든 메시지는 KU_A 로 암호화된다. 마지막 과정으로는 A 는 N_B 를 K_S 로 암호화해서 B 로 전송한다. 이 마지막 메시지를 통해서 A 가 K_S 를 가지고 있다는 사실을 B 에게 확인시켜준다[6,7].

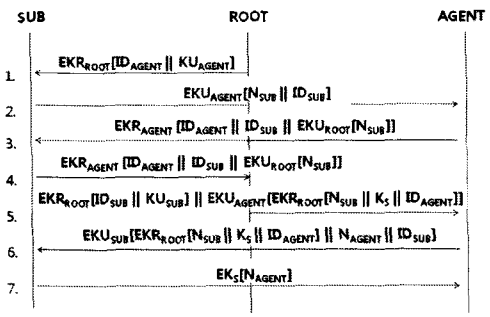
4. 안전한 패치 분배 시스템을 위한 효과적인 인증 구조

공개키 방식의 키 교환 기법을 사용하면 상호 간의 인증 뿐아니라 개인키로 암호화하고 복호화하기 때문에 부인 방지할 수 있다. 이러한 이유 때문에 계층적 구조를 가진 패치 분배 시스템의 인증 구조로 공개키 방식의 인증 구조를 채택한다. 하지만 본 논문의 계층적 패치 분배 구조 부분에서 언급했듯이 패치 분배 시스템은 두 가지 제약조건을 가진다. 첫 번째 제약조건은 중앙 집중형 인증 방식을 사용한다는 것이다. 두 번째는 하위 계층은 상위 계층의 IP 주소를 저장하지 않기 때문에 상위 계층으로 인증 요청을 할 수 없다는 것이다. 때문에 위에서 제시한 방식을 그대로 적용할 수 없다. 그래서 본 논문에서는 새로운 방식을 제안한다.

<표 3> 제안된 방식 표기법

SUB	서브 패치 관리 시스템
AGENT	에이전트
ROOT	루트 패치 관리 시스템
ID_{SUB}	서브 패치 관리 시스템의 신원
ID_{AGENT}	에이전트의 신원
N_{SUB}	서브 패치 관리 시스템이 생성한 임의의 값
N_{AGENT}	에이전트가 생성한

	임의의 값
KU_{ROOT}	루트 패치 관리 시스템의 공개키
KR_{ROOT}	루트 패치 관리 시스템의 개인키
KU_{SUB}	서브 패치 관리 시스템의 공개키
KR_{SUB}	서브 패치 관리 시스템의 개인키
KU_{AGENT}	에이전트의 공개키
KR_{AGENT}	에이전트의 개인키
K_S	서브 패치 관리 시스템과 에이전트의 세션키
E	암호화



[그림 4] 패치 분배 구조의 인증 절차

제안한 공개키 방식 인증 구조에서 ROOT, SUB, AGENT 는 기존의 공개키 방식 인증 구조 KDC, A, B 에 대응한다. 기존의 공개키 방식 인증 구조에서 문제가 되는 부분은 1 번 과정과 4 번 과정이다. 두 과정 모두 처음 연결을 시도할 때 하위 계층에서 상위 계층으로 연결 요청을 한다. KDC 역할을 하는 ROOT 가 인증을 진행할 SUB, AGENT 를 이미 알고 있기 때문에 기존의 1 번 과정을 생략한다. 기존의 4 번 과정은 반드시 필요한 과정이기 때문에 새롭게 제안한다. 우선 AGENT 는 ROOT 로 직접적인 인증 요청을 할 수 없다. 그렇기 때문에 일회적인 신뢰 관계에 있던 SUB 를 거쳐서 메시지를 ROOT 로 전송한다. 이 부분은 제안한 3 번 과정과 4 번 과정에 해당된다. 3 번 과정에서 AGENT 가 SUB 로 메시지를 보낼 때 KR_{AGENT} 로 암호화를 한다. 4 번 과정은 SUB 가 AGENT 에서 받은 메시지를 그대로 ROOT 에 전송한다. KR_{AGENT} 로 암호화함으로써 ROOT 는 AGENT 의 메시지를 인증한다. 위의 과정을 진행함으로써 상호 인증을 수행하고 안전하게 세션키를 교환한다.

5. 결론

본 논문에서는 대규모 네트워크 환경에서 패치 파일을 분배하기 위하여 계층적인 구조를 가진 패치 분배 시스템을 구성하였다. 또한 본 논문에서 구성된 제약 조건을 가진 패치 분배 시스템 환경에서 효과적으로 상호 인증을 하기 위한 인증 구조를 제안하였다. 제안한 인증 구조는 이미 검증된 공개키 기반의 인증 구조를 변경한 것이다.

추후 이러한 계층적 구조를 가지는 패치 분배 시스템을 실제로 구축하고 본 논문에서 제안한 인증 구조를 검증할 수 있는 체계를 구성해야 한다.

6. 참고 문헌

[1] 김윤주외 "확장성을 고려한 계층적 패치 분배 시스템 프레임워크 설계", 한국정보과학회 춘계학술 발표논문집 제 31 권 제 1 호(A), 2004. 4.

[2] 서정택외 "멀티플랫폼을 지원하는 패치 자동 관리 시스템", 한국정보과학회 추계학술발표논문집 제 30 권 제 2 호, 2003. 10.

[3] Miller, S. P., and Neuman, B. C., and Schiller, J. I., and Saltzer, J. H., "Section E.2.1: Kerberos Authentication and Authorization System", M.I.T. Project Athena, Cambridge, Massachusetts, December 1987.

[4] William Stallings, "Cryptography and Network Security Third Edition", Prentice Hall, 2003, pp384-385.

[5] Neuman, B., and Stubblebine, S. "A Note on the Use of Timestamps as Nonces." Operating Systems Review, April 1993.

[6] Woo, T., and Lam, S. "Authentication for Distributed Systems." IEEE Computer, January 1992.

[7] Woo, T., and Lam, S. "'Authentication'Revisited." IEEE Computer, April 1992.