Windows 환경에서의 의존성 해결을 위한 보안패치 분배 시스템 설계 및 구현

이상원°, 김윤주*, 손태식*, 문종섭*, 서정택**, 박응기**
고려대학교 정보보호대학원*, 국가보안기술연구소**
{a770720°, zzuya99, 743zh2k, jsmoon}*@korea.ac.kr, {seojt, ekpark}** @etri.re.kr

Design and Implement the Security Patch Distribution System for Resolving the Relation at Windows

Sangwon Lee°, Yun-Ju Kim*, Tae-Shik Sohn*, Jong-Sub Moon*, Jung-Taek Seo**, Eung-Ki Park**
Center for Information Security Technologies (CIST), Korea University*,
National Security Research Institute**

요 약

최근 네트워크를 통한 시스템 침해사고가 증가하고 있고, 이러한 침해사고는 대부분 시스템에 존재하는 취약성을 이용한 공격이므로 관련 패치의 설치는 매우 중요하다. 그래서 최근 자동화된 패치 관리 시스템의 연구가 많이 이루어지고 있다. 특히 시스템의 가장 기본적인 기능인 정확한 보안패치 분배는 중요하지만 벤더에 의존적이어야 하므로 어려움도 따른다. 본 논문에서는 Windows 환경에서의 의존성 해결을 위한 보안패치 분배 시스템을 설계하고 구현하고자 한다.

1. 서 론

사실 여러 명의 다른 프로그래머가 작성한 수백만 줄의 코드로 이루어진 운영체제와 응용 프로그램이 항상 안정적으로 동작한다는 것은 현실적으로 불가능하다. 이러한 점을 이용하여 공격자는 지속적으로 소프트웨어의 취약한 부분을 찾아내기 때문에 모든 공격을 예측한다는 것 역시 불가능한 일이다. 현실적인 대안은 소프트웨어 회사들이 제품 출시 이후에 드러나는 코드상의 취약점을 해결하기 위해서 내놓는 보안패치를 설치하는 것이다. 그러나, 일정 규모 이상의 조직에서는 이처럼 중요한 보안패치 관리를 위한 일련의 모든 작업들을 사용자에게만 맡길 수는 없기 때문에 중앙 집중적인 방식의 관리 작업이 이루어져야만 한다.

하지만 이러한 작업을 단순히 수동적인 방식으로 접근한다는 것은 매우 비효율적이다. 1인당 1대 패치에 걸리는 시간이 10분, 패치 실패율이 1%, 복구에 걸리는 시간이 1시간, 패치 대상 서버 수가 200대라면 한 사람이 하루 8시간씩의 작업으로 약 4일 간을 패치에 소요해야 한다는 결론을 얻을 수 있다[1]. 게다가 해마다 보고되는 보안 취약점은 크게 늘어나고 있으며, 이를 보완하기 위한 보안패치 역시 릴리즈 되는 시간 간격이 짧아지고 있는 실정이다[2]. 따라서, 효과적인 보안패치 분배 작업이 이루어지기 위해서는 자동화된 보안패치 관리 시스템이 필요하다.

그러나 이처럼 중요한 역할을 담당하고 있는 보안패치 관리 시스템이 적절하게 보안패치를 분배하지 못하거나 패치간의 의존성의 문제로 시스템에 악영향을 미칠 경우 오히려 조직의 보안성에 문제가 될 수 있다. 그러므로 의존성을 해결한다는 것은 보안패치 관리 시스템이 갖추어야 할 필수 요건이라고 할 수 있다.

2. 기존 보안패치 분배 시스템의 구성 및 시나리오

2.1 보안패치 관리 시스템 구성

패치 관리 프레임워크는 상이한 시스템들로 구성되어 있는 대규모 네트워크 환경에 적합한 보안패치를 자동 분배, 설치하는 시스템이다[3][4]. 그림 1은 패치 관리 프레임워크의 전체 구성도이며, 보안패치 서버, 보안패치 클라이언트 에이전트, 보안패치 서버 매니저, 보안패치 클라이언트 매니저, 보안패치 DB, 보안패치 저장소로 구성된다.

- **보안패치 DB** : 보안패치 파일과 관리자와 클라이언트의 정보를 보관하며, 패치 수집은 관리자에 의해 수행됨
- **보안패치 서버** : 클라이언트에게 필요한 보안패치를 DB로부터 얻어 실제 분배과정을 수행함
- **보안패치 매니저** : DB의 구성정보와 서버를 관리하며, 웹기반의 UI를 이용하여 관리자에게 편의성을 제공함

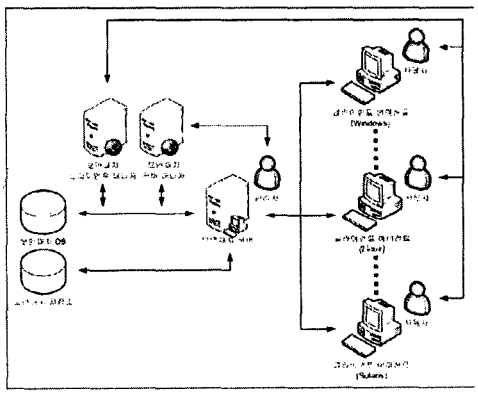

그림 1 보안패치 분배 시스템 구성도

- **보안패치 클라이언트** : 클라이언트 시스템의 정보 스캐닝과 패치 자동 설치를 수행함
- **보안패치 에이전트** : 클라이언트의 정보를 관리하며, 웹기반의 UI를 이용함.

2.2 보안패치 분배 및 설치

① 패치 서버에서 클라이언트에게 패치를 분배
- 패치를 설치 할 클라이언트 선별
- 패치 분배를 위한 클라이언트 인증
- 클라이언트 시스템에 패치 전송
- 패치 설치
- 클라이언트 프로파일 갱신

② 클라이언트에서 패치를 요청
- 클라이언트 프로파일 생성
- 패치 분배를 위한 클라이언트 인증
- 클라이언트 프로파일 전송
- 필요한 패치 검색
- 클라이언트 시스템에 패치 전송
- 클라이언트 프로파일 갱신

3. 기존 보안패치 분배 시스템의 한계

기존 보안패치 분배 시스템은 표 1과의 문제점들을 가지고 있었다.

표 1 기존 보안패치 분배 시스템의 문제점

문제점	설명
부정확한 패치 분배	클라이언트에 설치된 보안패치 정보를 얻어오는데 있어서, 설치된 보안패치 정보를 모두 얻어오지 못하여서 이미 설치된 보안패치를 설치해야할 대상으로 잘못 판단하는 경우가 발생하였고, 설치 대상의 구분이 단순히 이루어졌기 때문에 설치 대상에서 제외되는 경우에도 설치하도록 하여 문제가 존재하였다.
의존성 문제	보안패치가 설치되기에 앞서 설치되어 있어야 할 패치가 존재하더라도 그에 대한 고려를 하지 않았고, 서비스팩이나 누적패치가 배포되어 더 이상 설치하지 않아도 되는 패치에 대해서도 처리하지 않았다.

기존 프레임워크의 문제점들을 보완하여, 적절한 클라이언트에게 정확한 패치 분배, 의존성 문제를 해결함으로서 안정적인 보안패치 분배가 이루어지도록 해야 할 것이다.

4. 의존성 해결을 위한 보안패치 분배 시스템 설계 및 구현

의존성 해결을 위한 보안패치 분배 시스템은 클라이언트로부터 정확한 정보를 얻어오는 것과 보안패치 벤더에서 제공하는 보안패치 정보의 분석을 통해 클라이언트에게 적절한 보안패치를 분배할 수 있도록 하며, 사전에 설치되어 있어야 할 패치가 존재하는 보안패치의 경우에 적절히 대응하여 보안패치간의 의존성을 해결할 것이다.

4.1보안패치 벤더에서의 패치 배포 정보 분석

클라이언트에게 적절한 보안패치를 제공하기 위해 벤더에서 보안패치를 배포할 때 제공되는 배포 정보를 분석한다. 벤더에서 표 2와 같이[5] 정보를 제공한다. 이 정보들을 이용하여 사용자의 시스템에 적합한 패치를 선정하고, 사용자의 편의성을 위해 한번에 여러 개의 패치를 설치할 수 있도록 지원한다. 또한, 패치 설치 확인 정보를 통하여 프로파일 생성을 보다 정확하게 할 수 있으며 개인적으로 보안패치를 설치한 경우도 알아낼 수 있어서 중복되어 패치를 설치하는 경우를 막을 수 있다.

표 2 보안패치 벤더에서의 패치 배포 정보 분석

설치 플랫폼	보안패치가 설치되어야 할 설치 플랫폼이나 요구사항
설치 정보	다양한 설치 옵션을 제공
배포 정보	배포를 위해 무인모드로 재부팅 없이 설치할 수 있도록 설치 옵션을 제공
다시 시작 요구 사항	보안패치 설치 후 시스템 다시 시작이 필요성에 대한 정보 제공
제거 정보	제거 가능한지 가능하다면 어떠한 방법으로 가능한지 정보 제공
보안패치 대체 패치	이 보안패치가 기존에 제공된 보안패치를 대체하는 패치인지의 정보를 제공
파일 정보	보안패치 설치 후 변화된 시스템의 주요 파일의 정보를 제공
패치 설치 확인 정보	보안패치 설치 후 정상적으로 설치되었는지 확인하는 방법을 제공

4.2 클라이언트 프로파일 구성

클라이언트 프로파일은 클라이언트 시스템 정보와 시스템에 설치된 보안패치 정보로 구성된다. 클라이언트 시스템에 설치된 보안패치 정보를 얻기 위해 다음과 같이 세 가지 방식을 제안한다.

4.2.1 클라이언트 시스템의 레지스트리 이용

₩HKEY_LOCAL_MACHINE₩SOFTWARE₩Microsoft₩Updates에서 설치된 패치 정보를 얻어온다. 대부분의 보안패치가 설치되면 위의 레지스트리에 정보를 남기기 때문에 설치된 패치 정보를 얻을 수 있다.

- 장점 : 에이전트가 프로파일을 전송하면 서버는 필요한 패치를 검색하여 전송하므로 서버 클라이언트간의 통신횟수가 다른 방법과 비교하여 적다.
- 단점 : 특정 레지스트리에 정보를 남기지 않는 업데이트가 있기 때문에 일부 패치는 설치되었는지 확인할 수 없다. 그래서 모든 패치에 대해 서비스를 제공할 수 없다.

4.2.2 보안패치 설치 확인 정보 이용

4.1에서와 같이 MS에서 패치가 정상적으로 설치되었는지 확인하는 정보를 제공한다. 서버가 DB에 저장되어있는 모든 보안패치의 설치 확인 정보를 에이전트에게 전달하여 각 패치들이 설치되었는지 확인하여 설치된 보안패치의 정보로 프로파일을 생성한다.

- 장점 : 설치된 패치 정보를 얻어 설치를 확인하기 때문에 클라이언트에 설치된 패치 정보를 정확히 알 수 있다. 따라서 보안패치 분배의 정확성을 기대할 수 있다.
- 단점 : DB에 저장된 모든 보안패치에 대해 설치를 확인하기 때문에 클라이언트 에이전트에 과부하가 발생할 수 있다. 또한 서버와 클라이언트간에 주고받아야 할 정보가 많아져 네트워크 트래픽을 증가시킨다.

4.2.3 레지스트리와 보안패치 설치 확인 정보 이용

대부분의 보안패치는 특정 레지스트리에 정보를 남기므로 이 레지스트리에 있는 정보로 클라이언트 프로파일을 생성하여 서버에게 전송한다. 서버는 프로파일의 정보로 클라이언트에 적용되어야 할 보안패치들을 선택하고 그 패치들에 대한 설치 확인 정보를 에이전트에게 전송한다. 클라이언트는 각 패치가 설치되었는지 확인하고, 설치되지 않은 패치를 서버에게서 분배받는다.

- 장점 : 4.2.1의 방안과 4.2.2의 방안을 결합하여, 클라이언트에서 보안패치 설치를 확인하는 경우의 수를 줄였기 때문에 클라이언트의 부하를 줄일 수 있고, 예외 없이 설치된 패치 정보를 얻을 수 있다. 또한 클라이언트에게 필요한 패치에 대한 설치 확인 정보를 전송하므로 네트워크 트래픽도 감소시킨다.

4.2.4 적용 방안

4.2.3의 방안이 가장 효율적이기 때문에 보안패치 프레임워크에 적용하겠다. 특정 레지스트리의 정보만으로 프로파일을 생성하면 위의 레지스트리에 정보를 남기지 않는 보안패치의 경우 프로파일에 적용되지 않으므로 설치 후에도 서버가 알 수 없다. 그래서 이런 패치의 설치 정보를 포함하는 또 다른 프로파일을 그림 2와 같이 생성한다. 그 프로파일 이름은 profile2이며, 보안패치 서버로부터 보안패치 목록과 설치 확인 정보를 받으면 각 보안패치의 설치 여부를 결정하고 설치된 패치의 정보와 설치 확인 정보를 profile2에 기록한다.

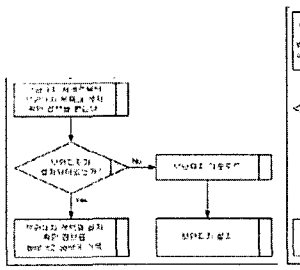

그림 2 profile2 생성 과정

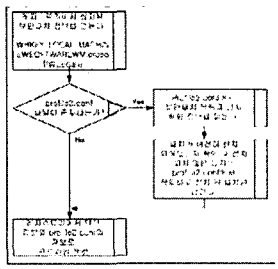

그림 3 프로파일 생성 과정

그리고 그림 3과 같이 서버에게 전송되는 프로파일을 생성할 때 기본적으로 레지스트리를 검색하여 설치된 패치의 정보를 얻고, profile2에 기록된 보안패치가 제거되었을 가능성을 고려하여 여전히 설치 되어있는지를 다시 확인한 후 설치되어 있는 패치 정보만으로 프로파일을 생성한다.

4.3 보안패치 DB 구성

보안패치 DB는 클라이언트 개인신상 정보, 클라이언트 시스템 정보, 보안패치 정보를 저장하고 있다. 클라이언트 시스템 정보는 클라이언트 시스템의 운영체제 정보, 네트워크 정보, 보안패치 분배에 필요한 각종 프로그램의 버전 정보 등을 포함한다. 그리고 보안패치 정보는 벤더에서 제공하는 보안패치 관련 정보들과 서버에 실제 패치파일이 존재하는 위치, 각 보안패치를 구분하는 정보 등을 포함한다. 클라이언트에게 서비스팩도 제공하기 위해 서비스팩 관련 정보들을 보안패치 정보와 구별하여 저장한다.

4.4 보안패치 검색 방안

보안패가 분배되는 경우는 클라이언트가 서버에 프로파일을 전송하여 필요한 패치가 검색된 경우와 새로운 패치가 등록되어 분배받아야 할 대상 클라이언트를 검색하여 분배하는 경우로 나눌 수 있다.

4.4.1 클라이언트 프로파일을 이용한 보안패치 검색

우선 클라이언트 시스템에 최신 서비스팩이 설치되어있는지 검사한다. 서비스팩 검색은 DB에서 동일한 플랫폼용 최신 서비스팩 번호

를 얻어와, 현재 클라이언트 시스템의 서비스팩 번호를 비교하여 최신 서비스팩이 설치되어 있는지 검사한다. 클라이언트 시스템에 최신 서비스팩이 설치되어 있지 않은 경우 서비스팩은 개별 설치를 요구하기 때문에 보안패치 분배에 앞서 분배한다.

그리고 클라이언트에 설치되어야 할 보안패치를 검색한다. 그림 4와 같이 클라이언트 프로파일에서 플랫폼 정보를 얻어와 클라이언트 시스템에 적용되어야 할 보안패치를 DB에서 검색한다.

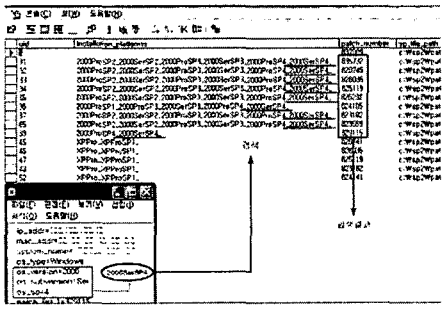

그림 4 보안패치 검색

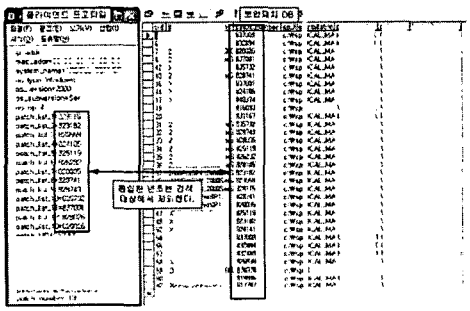

그림 5 이미 설치된 보안패치 검색대상에서 제외

또한 그림 5처럼 이미 클라이언트 시스템에 설치된 패치를 다시 분배하지 않도록 설치된 패치를 검색대상에서 제외시킨다. 이미 설치된 패치 정보는 클라이언트 프로파일을 통해서 알 수 있다.

4.4.2 새로운 보안패치에 대한 대상 클라이언트 검색

새로운 보안패치가 서버에 등록되었을 때 대상이 되는 클라이언트를 검색한다. 서비스팩이 등록되는 경우와 보안패치가 등록되는 경우로 나눌 수 있는데 우선 서비스팩이 등록되는 경우에는 DB에 저장되어 있는 서비스팩의 적용대상 운영체제 정보로 클라이언트 시스템 정보를 포함하는 DB에서 대상 클라이언트를 검색한다.

그리고 보안패치가 등록된 경우는 그림 6과 같이 보안패치의 적용대상이 되는 운영체제 정보와 서비스팩 정보를 이용하여 서비스팩의 경우와 마찬가지로 클라이언트 시스템 정보를 포함하는 DB에서 대상 클라이언트를 검색한다.

4.7 보안패치 설치 방안

그림 7과 같이 서버로부터 보안패치 관련 정보를 전송 받으면 서비스팩인지 보안패치인지 구분한다. 보안패치이면 설치 확인 정보가 서버로부터 받은 정보에 포함되어 있을 것이다. 이 정보를 이용하여 이미 설치되어있는지 검사하고 설치되어 있다면 profile2에 기록한다. 그리고 설치대상에서 제외한다. 이것은 벤더에서 제공하는 보안패치 관리모듈에 의해 패치가 설치되었거나 사용자가 직접 설치했을 경우에도 다시 설치되는 일이 없도록 할 수 있다. 기존에 설치가 되어 있지 않았다면 설치해야할 보안패치를 설치하기 전에 설치되어 있어야 할 패치가 존재하여 설치되어 있지는 않은지 검사한다. 설치되어 있지 않다면 역시 설치대상에서 제외한다. 처음과정 설치대상에서 제외되겠지만 다음 과정에서는 사전에 설치되어 있어야 할 패치가 전 과정에서 설치되기 때문에 설치될 것이다. 이 과정을 설치해야할 패치의 개수만큼 반복하여 최종적으로 설치해야할 대상 패치만을 서버로부터 다운로드받고 설치한다.

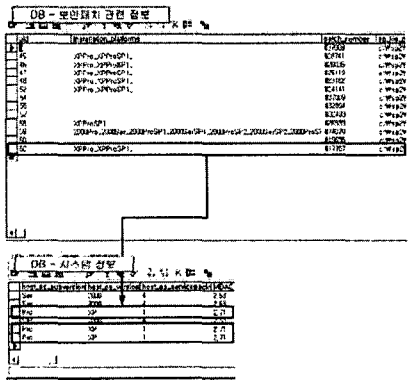

그림 6 대상 클라이언트 검색

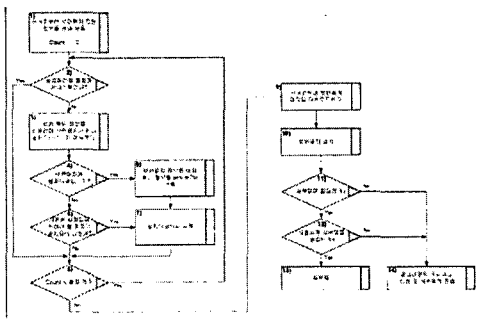

그림 7 보안패치 설치 과정

5. 결 론

본 논문에서는 Windows환경에서의 의존성 해결을 위한 보안패치 분배 시스템을 설계하는 과정에서 의존성 해결을 위한 방안을 모색, 개발하였다.

이러한 내용들을 기반으로 운영체제만이 아니라 클라이언트 시스템의 안정성을 해칠 수 있는 주요 응용 프로그램들에 대한 보안패치 분배 방안, 보안패치 설치 이후에 이상 동작 여부를 확인할 수 있는 로그 기능, 로그 기능을 기반으로 하여 해당 보안패치를 설치하기 이전의 상태로 되돌려 줄 수 있는 롤백 기능 등 보안패치 분배 시스템의 안정성을 확보를 위하여 다양한 분야에 대한 연구가 이루어져야 할 것이다.

6. 참고 문헌

[1] 이용학, "보안 시스템 관리 방안 - 패치 매니지먼트 시스템", *NETWORK TIMES*, March 2004
[2] CERT Coordination Center, http://www.cert.org
[2] 안철수연구소, http://home.ahnlab.com/
[3] Sohn Tae-Shik, "Safe Patch Distribution Architecture in Intranet Environments", SAM 2003
[4] Cheol-Won Lee, "A Secure Patch Distribution Architecture", Intelligent Systems Design and Applications, A. Abraham Eds, Advances in Soft Computing, Springer-Verlag, pp 229-238
[5] Microsoft Corporation, http://www.microsoft.com