확장성을 고려한 계층적 패치 분배 시스템 프레임워크 설계

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

Design the Classed Patch Distribution System Framework Considering the Extension

YunJu Kim° ,SangWon Lee*, Tae-Shik Sohn*, Jong-Sub Moon*, JungTaek Seo**, JuBum Yun**, EungKi Park**
Center for Information Security Technologies (CIST), Korea University*,
National Security Research Institute**

요 약

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. 하지만 패치 관리의 대상이 되는 기업, 공공 기관 등과 같이 대규모 네트워크를 구축하고 있는 그룹은 그 규모와 구조가 유동적일 수 있다. 그러므로 다양한 환경에서도 무리 없이 패치 분배 서비스를 지원하기 위해서는 패치 관리 대상 그룹의 확장성을 고려해야만 한다. 본 논문에서는 확장성을 고려한 계층적 패치 분배 시스템의 프레임워크를 제시한다.

1. 서 론

2003년 1.25 인터넷 대란을 일으킨 SQL Slammer worm은 6개월 전에 제공된 보안패치를 설치하였다면 피할 수 있었다. 하지만 전 세계적으로 대란을 겪은 것은 신속하게 패치를 분배, 적용하지 못했음을 의미한다. 소프트웨어의 취약점을 이용한 공격은 패치를 적용하는 것만으로도 막을 수 있기 때문에 패치의 적용은 매우 중요하다.

하지만 현실적으로 새로운 취약점과 패치가 빠른 속도로 등장하기 때문에 수동적으로 패치를 관리한다는 것은 어려움이 따른다. 예를 들어 수천대의 이기종 서버와 PC를 관리해야 할 경우, 수동적으로 패치를 적용한다면 각종 패치를 수집, 테스트, 배포, 적용, 확인하는 시간과 비용을 수반할 것이며, 정확성도 기대할 수 없다. 그래서 최근 자동화된 패치 관리 솔루션이 이슈가 되고 있다.

그러나 패치 관리 대상이 되는 기업, 공공 기관 등과 같이 대규모 네트워크를 구축하고 있는 그룹은 그 규모와 구조가 유동적일 수 있다. 이를테면, 지역별로 각각 다른 서버를 운영하되 효율성을 높이기 위해서 이에 대한 관리만은 통합하고자 하거나, 서로 다른 그룹을 이루고 있는 네트워크를 특정한 필요에 의해서 하나의 그룹으로 통합하고자 할 때, 그리고 클라이언트의 수가 정상적인 패치 분배 서비스를 제공할 수 없을 정도로 많아져 서버를 증설해야하는 경우 등이 발생할 수 있다. 이처럼 다양한 환경에서도 무리 없이 패치 분배 서비스를 지원하기 위해서는 패치 관리 대상 그룹의 확장성을 고려해야만 한다. 그러므로 본 논문에서는 확장성을 고려한 계층적 패치 분배 시스템의 프레임워크를 제안하도록 하겠다.

2. 보안패치 관리 프레임워크

2.1 보안패치 관리 시스템 전체 구성

패치 관리 프레임워크는 상이한 시스템들로 구성되어 있는 대규모 네트워크 환경에 적합한 보안패치를 자동 분배, 설치하는 시스템이다[1, 2]. 그림1은 패치 관리 프레임워크의 전체 구성도이며, 보안패치 DB, 보안패치 서버, 보안패치 매니저, 보안패치 클라이언트, 보안패치 에이전트로 구성된다.

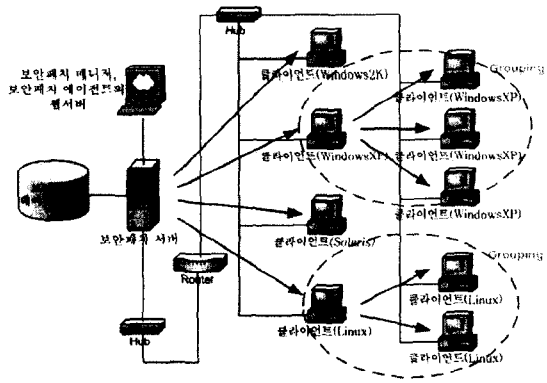

그림 1 프레임워크 전체 구성도

- **보안패치 DB** : 보안패치 파일과 관리자와 클라이언트의 정보를 보관하며, 패치 수집은 관리자에 의해 수행됨
- **보안패치 서버** : 클라이언트에게 필요한 보안패치를 DB로부터 얻어 실제 분배과정 수행함
- **보안패치 매니저** : DB의 구성정보와 서버를 관리하며, 보안패치 매니저는 웹기반의 UI를 이용하여 관리자에게 편의성을 제공함
- **보안패치 클라이언트** : 대상은 Windows 2000, XP, Solaris, Redhat 시스템들이며, 클라이언트 시스템의 정보 스캐닝과 패치 자동 설치를 수행함
- **보안패치 에이전트** : 클라이언트의 정보를 관리하며, 보안패치 에이전트는 웹기반의 UI를 이용함.

2.2 보안패치 분배 및 설치

패치를 클라이언트 시스템에 분배하는 방법은 패치 서버에서 패치를 분배하는 경우와 클라이언트에서 패치를 요청하는 경우가 있다.

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

2.3 그룹화를 이용한 보안패치 분배

대규모 네트워크에 구성되어 있는 클라이언트들에 대해서 패치 서버가 1:N 방식으로 분배하는데 따르는 비효율성을 보완하기 위해 그룹화를 이용한 트리기반의 분배 구조를 갖는다.

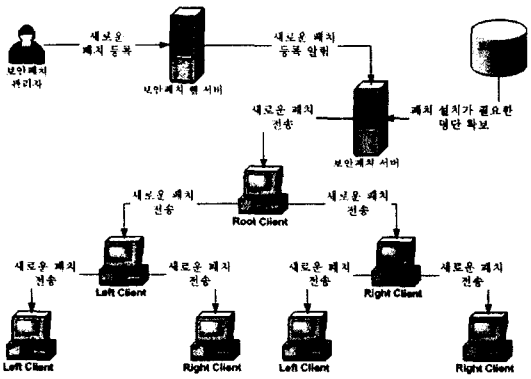

그림 2 그룹화에 따른 패치 분배 과정

그림2는 그룹화에 따른 패치 분배 과정이다. 클라이언트에 대한 그룹화가 필요한 경우는 패치 DB에 새로운 패치가 업데이트 되는 경우이다. 이때 패치 서버는 업데이트 된 패치에 대하여 설치가 필요한 클라이언트들을 검색하고 하나이상의 클라이언트가 검색되면 검색된 클라이언트들을 하나의 그룹으로 묶고 클라이언트 중 대표 클라이언트를 선정하여 그 클라이언트에게 같은 그룹에 속하는 그룹 리스트와 업데이트 된 패치를 전송한다. 패치를 전송 받은 대표 클라이언트는 그룹 리스트 에 포함되어 있는 클라이언트들에게 다시 이 패치를 전송한다 [3].

3. 보안패치 관리 프레임워크의 한계

보안패치 서버의 부하를 줄이기 위해 제안된 그룹화에 따른 트리형 분배 구조는 클라이언트가 다른 클라이언트에게 패치를 전달하는 중간 서버의 역할을 해야한다. 이는 개인 프라이버시의 관점에서 문제가 있다. 사용자는 자신의 시스템이 분배서버로 이용되는 것을 원하지 않을 수 있기 때문이다. 따라서 비효율적이라는 문제점이 있지만 중앙 서버방식을 채택할 수밖에 없다.

대규모 네트워크를 대상으로 하는 보안패치 관리 시스템이 중앙 서버방식을 사용하지만 서버에 부하를 줄일 수 있는 방안으로, 그리고 서론에서 언급한 것과 같이 다양한 환경에서 분배 서비스를 제공하기 위해 다음과 같이 확장성을 고려한 계층적 패치 분배 방안을 제시한다.

4. 확장성을 고려한 계층적 패치 분배

4.1 서로 다른 그룹을 통합시키는 경우(DB 분리)

서로 다른 두 개의 그룹을 하나로 통합하여 패치 분배 서비스를 제공하는 경우, 우선 각각의 서버 매니저에서 관리하던 회원 DB를 어떻게 처리할 것인가에 대한 문제에 대한 해결책이 필요하다.

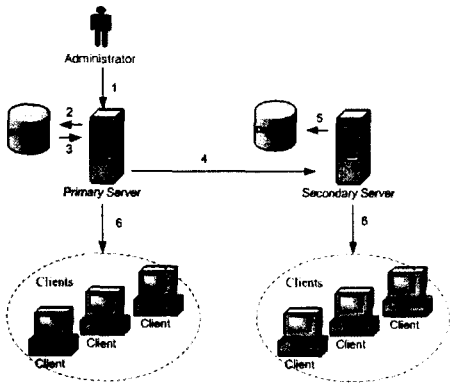

그림 3 서로 다른 그룹을 통합시키는 경우

하나로 통합하려는 두 개의 그룹이 멀리 떨어져 있다거나 패치 분배 관리상의 문제로 통합은 했지만 회원 DB까지 통합할 필요가 없는 등 회원 DB를 각각의 서버에서 통합 전처럼 따로 보관·관리할 필요가 있을 수 있다. 즉, 관리만 통합하는 경우이다. 이러한 경우에는 통합하려는 서버 중에서 서버 관리자가 직접 패치를 등록하고, 환경을 설정하는 서버를 Primary Server로 설정하도록 한다. Primary Server는 자신의 밑으로 하위 Server들이 존재할 수 있는데 그 서버들을 Secondary Server라고 한다. DB에 Secondary Server의 정보를 저장하는 테이블을 구성하고, 이 DB를 이용하여 Primary Server가 하위에 Secondary Server가 존재하는지를 검색할 수 있다. 그림3에서 보면 (1) 패치 관리자가 새로운 패치를 등록하면 (2) Primary Server의 DB에 패치 정보를 저장하고 (3) DB를 검색하여 Secondary Server의 존재를 검사한다. 검색하여 존재할 경우 (4) Secondary Server로 패치 파일과 패치 정보를 전송하고, 동시에 서버가 관리자로부터 패치를 등록 받을 때와 같이 새로운 패치가 등록된 사실을 알려준다. secondary server는 자신의 관리자가 패치를 등록한 것으로 인식하고 primary Server와 동일한 작업을 진행하게 된다. (5) 받은 패치 정보를 DB에 저장함으로써 Primary Server와 동일한 환경을 구축한다. (6)

Primary Server와 Secondary Server는 각각 자신의 클라이언트에게 패치를 분배하여 일련의 과정을 수행한다.

4.2 서로 다른 그룹을 통합시키는 경우(DB 통합)

4.1의 경우와 반대로 회원 DB까지 통합하여 관리해야 하는 경우가 생길 수 있다. 이러한 경우에는 관리자의 편의를 위해서 자동으로 회원 DB를 통합하여 주는 환경을 제공하며 이렇게 통합된 환경에서는 결국 하나의 서버에 여러 개의 클라이언트가 존재하는 형태가 되기 때문에 별도의 작업은 필요치 않을 것으로 판단된다. 하지만 회원 DB를 통합할 때 클라이언트의 고유한 정보인 사용자 ID가 중복되는 경우가 발생할 수 있다. 그래서 DB에 서버와 관련된 필드를 두고, 서버의 고유한 정보와 함께 보관하고, 각 클라이언트가 서버에 등록할 때 서버의 정보를 함께 얻을 수 있도록 함으로써 ID 중복 문제를 해결할 수 있다. 그림4는 DB통합시 ID 중복문제를 해결하기 위해 서버 관련 필드를 사용한 DB의 예이다. 동일한 ID인 "efgh"를 서버의 정보로 구별하는 것을 볼 수 있다.

Server1의 DB

UserID	ServerInfo	.....
abcd	server1	.....
efgh	server1	.....
.....	.....	.....

Server2의 DB

UserID	ServerInfo	.....
efgh	server2	.....
ijkl	server2	.....
.....	.....	.....

Server1과 Server2의 통합된 DB

UserID	ServerInfo	.....
abcd	server1	.....
efgh	server1	.....
efgh	server2	.....
ijkl	server2	.....
.....	.....	.....

그림 4 패치 DB 통합 예제

4.3 패치 분배를 위한 서버를 중설하는 경우

중앙 서버의 부하를 줄여 패치 분배의 효율성을 증대시키기 위해서 별도의 분배 서버를 운영하는 환경이 존재할 수 있다. 이 분배 서버를 Distribution Server라고 한다. 이때 다수의 패치 분배 서버를 효과적으로 운영하기 위해서는 관리자가 Distribution Server들의 정보를 DB에 등록하여 관리한다. 패치를 분배받아야 할 대상이 하나 이상일 경우 패치 서버가 각각의 Distribution Server들에게 패치 파일과 그들이 분배해 줄 클라이언트 대상 명단을 분할하여 나누어주도록 한다.

4.4 전체 구성

그림 5는 확장성을 고려한 계층적 분배 시스템의 전체 구성도이다. ①번 서버는 ②번 서버의 Primary가 되고 ②번 서버는 ①번 서버의 Secondary가 된다. 마찬가지로 ②번 서버는 ③번 서버의 Primary가 된다. 즉, ②번 서버는 ①번 서버의 Secondary인 동시에 ③번 서버의 Primary인 것이다. 이런 구조로 한 번의 관리로 여러 그룹의 분배 서비스를 제어할 수 있게 된다. 또한 서버의 부하를 줄이기 위해 ④번과 같은 Distribution Server를 둘 수 있다.

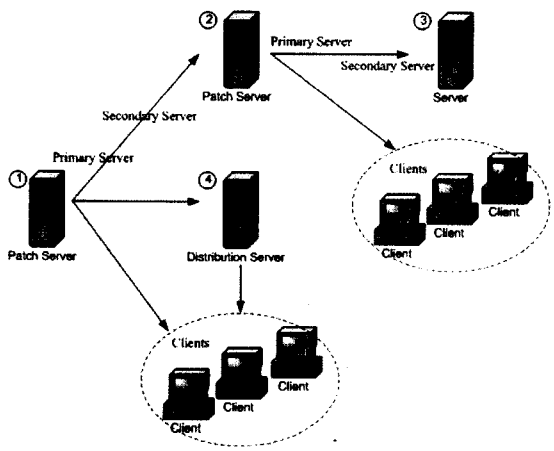

그림 5 확장성을 고려한 계층적 분배 시스템의 전체 구성도

5. 결 론

최근 자동화된 패치 관리 시스템의 연구가 많이 이루어지고 있지만 패치 관리 대상이 되는 대규모 네트워크 그룹의 규모와 구조가 유동적임을 간과하고 있다. 환경이 변화되더라도 정상적인 패치 분배 서비스를 제공하는 것이 필요할 것이다.

본 논문에서 제안하는 프레임워크로 패치 분배 서비스를 제공하는 것은 하나의 서버를 관리함으로서 여러 그룹을 동시에 제어할 수 있게 된다. 이는 다양한 벤더에 산재되어 있는 패치를 수집하고, 테스트, 배포, 적용, 확인하는 시간과 비용을 줄일 수 있고, 개인 프라이버시의 문제를 갖고 있는 그룹화를 사용하지 않고 서버의 부하를 줄일 수 있는 장점이 있다.

6. 참고 문헌

[1] Sohn Tae-Shik, "Safe Patch Distribution Architecture in Intranet Environments", SAM, 2003

[2] Cheol-Won Lee, "A Secure Patch Distribution Architecture", ISDA 2003, Lecture Notes in Computer Science, Springer-Verlag, 2003

[3] G. Caronni and M. Waldyogel, "Efficient Security for Large and Dynamic Multicast Groups", (WETICE 98) IEEE Comp Society Press, 1998

[4] "Vulnerabilities in Operating-System Patch Distribution", http://razor.bindvie-w.com/publish/papers/os-patch.html