시스템 보안성 향상을 위한 패치관리시스템 설계 및 구현

서정택* · 윤주범* · 최대식* · 박응기* · 박춘식*

요 약

운영체제 및 응용프로그램들은 프로그램 개발 과정의 특성상 보안 취약성을 가지고 있다. 최근 이러한 취약성을 악용하는 침해사례가 증가하고 있으며, 그 피해의 파급효과가 더욱 커지고 있다. 패치의 분배 및 설치는 취약성을 이용하는 침해사고를 예방하기 위한 가장 중요한 요소 중의 하나이다. 특정 기관이나 조직은 다양한 운영체제 및 응용프로그램을 사용하기 때문에 관리자가 매번 신속하게 모든 시스템들에 대하여 패치를 설치하기는 어려움이 있다. 본 논문에서는 중앙의 관리자가 패치관리서버를 이용하여 Windows, Linux, Solaris 클라이언트 시스템들에 대하여 안전하게 패치를 자동분배하고 설치하는 패치관리시스템을 설계 및 구현하였다. 또한, 대규모 네트워크를 지원하기 위하여 확장성을 고려한 계층적인 패치 분배 구조로 설계 및 구현하였다.

Design and Implementation of Patch Management System for Improving System Security

Jung-Taek Seo* · Joo-Beom Yun* · Dae-Sik Choi*
Eung-Ki Park* · Choon-Sik Park*

ABSTRACT

Operating systems and application programs have security vulnerabilities derived from the software development process. Recently, incident cases related with the abuses of these vulnerabilities are increasing and the damages caused by them are becoming very important security issues all over the nations. Patch management is one of the most important processes to fix vulnerabilities of softwares and to ensure a security of systems. Since an institute or a company has distributed hierarchical and heterogeneous systems, it is not easy to update patches promptly. In this paper, we propose patch management framework to safely distribute and install the patches on Windows, Linux, and Solaris client systems. Besides, we considered extensibility and hierarchical structure for our patch management framework to support large scaled network environment.

Key words : Patch Management System, System Security, Operationg System

* 국가보안기술연구소

1. 서 론

일반적으로 모든 운영체제 및 응용 프로그램들은 프로그램 개발 과정의 특성상 보안 취약성을 가지고 있다. 이러한 보안 취약성을 악용하는 침해사례가 급증하고 있으며, 그 피해의 파급효과가 커지고 있다. 이러한 추세에서 패치에 대한 안전하고 신속한 분배 및 설치는 해당 시스템의 보안을 위한 가장 기본적이고 필수적인 요소로 강조되고 있다[1, 2].

하지만 시스템 관리자들이 패치 일일이 해당 사이트에 가서 패치를 다운받아야 부분과, 관리하고 있는 시스템이 이기종이고 그 수가 많은 경우에 효과적인 패치 관리에 어려움이 있다. 또한, 패치의 분배 및 설치 과정에서 패치 정보의 누출이나 패치를 가장한 트로이목마와 같은 백도어의 설치 등과 같은 보안상의 문제점을 가져올 수 있다[11, 14].

본 논문에서는 중앙의 패치 서버 프로그램이 각 벤더들로부터 패치를 다운받아 DB에 저장하고, 프로파일 관리기법을 이용하여 해당 패치를 필요로 하는 클라이언트시스템들을 선별하여 패치를 자동으로 분배하고, 설치하는 중앙 집중화된 패치관리시스템을 설계 및 구현한다. 또한, 대규모네트워크를 지원하기 위해서 확장성을 고려하여 계층적인 패치 분배 구조로 설계 및 구현하였다.

2. 동향분석

효과적인 패치관리시스템의 설계 및 구현을 위하여 국외 상용화 제품 세부 기능을 분석하고, 운영체제 벤더별로 패치 분배기술을 분석하였다. <표 1>은 국외 패치관리시스템의 세부 기능을 분석한 표이다. 세부 기능을 살펴보면, 우선 제품에 따라 에이전트에 기반 한 제품이 있고, 그렇지 않은 제품이 있다[5-9].

〈표 1〉 국외 상용화제품 세부기능 분석표

	본 연구	Patch Link	BigFix	Shavlik	Gravity Strom
Agent Based	O	O	O	×	×
Hierarchical Distribution	O	×	×	×	×
Multi platform	O	O	O	×	×
Client scanning	O	O	O	×	×
Secure Transfer	O	O	O	×	×
Patch support for user application	O	O	O	O	O
Group	O	O	×	×	O
Patch file Encryption	O	O	O	O	O

에이전트에 기반 하지 않는 시스템의 경우 에이전트를 배포하지 않아도 되므로, 배포가 용이하지만, 반면에 대상 시스템의 정보를 얻기 위해서는 직접적으로 네트워크를 통해 분석을 해야 하므로, 네트워크 트래픽이 많이 발생하는 단점이 있다. 본 연구에서는 배포의 편리성 보다는 정보 수집 및 네트워크 부하를 고려하여 에이전트 기반의 패치관리시스템으로 설계 및 구현하였다.

국외 상용 제품의 경우 하나의 패치분배 서버가 다수의 클라이언트들에 대하여 패치를 분배하고 있다. 이는 대규모 네트워크에 대한 지원이 어려우며, 패치 분배 시 과부하가 발생할 수 있는 문제점을 가지고 있다. 따라서, 본 연구에서는 확장성을 고려한 계층적 패치 분배구조를 지원하는 시스템으로 설계 및 구현하였다.

또한, 본 연구에서는 Windows, Linux, Solaris의 다양한 운영체제 환경에서 작동하고 있는 클라이언트 시스템들에 대한 보안패치 관리를 일률적으로 가능하도록 설계 및 구현하였다[10, 12, 13].

패치 분배 시 안전한 채널을 통하여 분배가 이루어지지 않으면, 패치 정보의 누출이나 패치를 가장한 트로이목마와 같은 백도어의 설치 등과 같은 보안상의 문제점이 발생할 수 있으므로 본 연구에서는 SSL을 이용하여 안전한 패치 분배를 수행하도록 한다.

그 밖에도 그룹화 기능은 관리 대상 시스템을 네트워크, 운영체제 등으로 그룹화 하여 관리할 수 있는 기능을 제공하게 되어, 효율적이고 편리한 패치 분배가 용이하도록 하고 있다.

<표 2>는 각 운영체제 벤더별 패치 분배 관련 기술을 분석한 내용이다.

〈표 2〉 운영체제 벤더별 패치분배 기술 분석표

구분	Microsoft (Windows)	Sun (Solaris)	RedHat (Linux)
패치 파일 형태	실행파일	PatchID 파일	RPM 파일
분배 방법	http, https 자동업데이트	http, ftp, https	http, ftp, https 자동업데이트
인증	Microsoft Secure Server Authority	Sun Microsystems Inc CA (Class B)	Verisign Secure Server Certification Authority
기밀성	SSL 128bit(옵션)	SSL 128bit(옵션)	SSL 128bit
무결성	SHA1 (IE 인증)	SHA1 (IE 인증)	SHA1(IE 인증)
패치 파일 설치 방법	Active X (자동)	patchadd, patch cluster	genie 자동 rpm 수동
특이 사항	• rollback 기능 • dministrator 권한 • 자동업데이트 시 기밀성 제공하지 않음	• 패치파일을 관리툴로 verify • root 권한 • 패치 의존성 여부	• 패치설치 DMZ 기능 • 자동업데이트는 rpm에 한함 • 패치 의존성 여부 • 자동업데이트 시 인증

각 운영체제 벤더별로 패치에 대한 업데이트 기능을 제공한다. 각 운영체제시스템마다 운영환경 및 패치파일의 형태가 각각 다르다. 윈도우의 경우 일반적인 Widows Installer 형태의 실행파일로 제공되며, Solaris의 경우 PatchID file, 그리고 Linux의 경우 RPM file로 패치가 제공된다. 패치의 분배 방법은 세 개의 운영체제 모두 비슷한데, 대부분 http나 https 혹은 자동업데이트 기능을 사용하고 있다. 인증의 경우에는 세 개의 운영체제가 전부 다른 방법을 사용하고 있다. Linux를 제외하고는 자사의 인증 솔루션을 사용하여 인증을 한다. Linux의 경우 Verisign에서 제공하는 인증 서버를 사용한다. 기밀성은 세 개의 운영체제 모두 SSL 128bit를 사용하는데, Linux만이 필수이고 나머지는 옵션으로 제공하고 있다는 것이 특징이다. 무결성은 세 개의 운영체제 모두 SHA1을 지원한다. 그 밖에 운영체제의 환경에 따라 각각 운영체제에 특성에 맞게 여러 가지 기능들을 제공하고 있다.

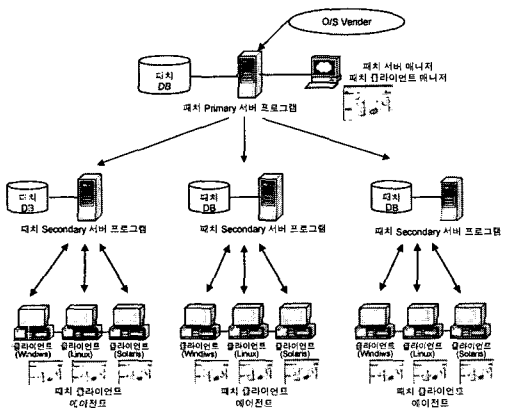
3. 제안하는 시스템

3.1 시스템 보안성 향상을 위한 패치관리시스템 전체 구성

시스템 보안성 향상을 위하여 대규모 네트워크를 대상으로 패치 관리를 제공해야 하는 경우 여러 대의 서버를 이용하게 되고 각 서버에 대한 별도의 관리가 요구되므로 서버의 개수만큼 패치의 수집, 배포, 테스트에 드는 비용을 감수해야한다. 그러나 계층적 패치 분배 기술을 이용하면 한 번의 관리로 여러 그룹의 클라이언트 시스템들에 대한 패치관리를 편리하게 효과적으로 수행 가능하고, 분배서버의 과부하도 줄일 수 있다[3, 4].

(그림 1)은 시스템 보안성 향상을 위하여 대규모 네트워크를 지원하는 계층적 패치관리시스템

의 전체구성도이다. 패치관리시스템은 패치 DB, 패치 서버 프로그램, 패치 서버 매니저, 패치 클라이언트 에이전트, 패치 클라이언트 매니저로 구성된다.



(그림 1) 시스템 전체구성도

- **패치 DB** : 보안패치 분배 및 설치를 위한 중요한 정보들을 보관하는 구성요소로서, 패치 파일과 관리자와 클라이언트의 정보를 저장하며, 패치 수집은 관리자에 의해 수행되거나, FTP 사이트로 패치를 제공하는 부분에 대해서는 자동 구성이 가능하다.
- **패치 서버 프로그램** : 패치 클라이언트 에이전트와 상호 통신작업을 통하여 각종 정보들을 주고받으며, 클라이언트 프로파일을 관리하고, 새로운 패치가 등록되면 설치가 필요한 클라이언트들을 선별하여 패치파일을 분배한다.
- **패치 서버 매니저** : 관리자가 패치관리시스템을 관리할 수 있도록 Web 인터페이스를 제공하는 구성요소로서, 각종 정보들을 추가, 수정 및 관리할 수 있도록 지원해준다.
- **패치 클라이언트 에이전트** : 서버 프로그램과 상호 통신작업을 통하여 각종 정보들을 주고받을 수 있도록 지원하는 구성요소로서, 초기에 설치하면 시스템에 대한 스캐닝을 실시하여 각종 정보들을 패치 서버 프로그램에 전송하고, 패치 서버 프로그램으로부터 분배 받은 패치를 자동적으로 클라이언트 시스템에 설치한다.
- **패치 클라이언트 매니저** : 클라이언트 시스템 사용자가 스스로 자신의 시스템에 대한 각종 정보들을 확인할 수 있도록 Web 인터페이스를 제공하는 구성요소로서, 자신에게 필요한 패치 목록을 조회할 수 있으며 직접 패치분배를 요청할 수 있다.

대규모 네트워크를 지원하기 위해서 (그림 1)에서와 같이 Primary 서버와 Secondary 서버로 구성하여 운영할 수 있다. 이 경우 관리자는 중앙의 Primary 서버만 관리하며 대규모 네트워크 내의 다수의 클라이언트 시스템들에 대하여 패치 관리를 수행할 수 있다.

3.2 패치 프로파일

패치 프로파일은 패치관리시스템의 운영 및 관리를 위해 아주 중요한 정보를 제공해주는 매개체로서 클라이언트 에이전트가 클라이언트 시스템에 대한 스캐닝을 실시하여 프로파일의 정보들을 만들어 낸다. 패치 프로파일에는 컴퓨터

〈표 3〉 패치 프로파일 프로토타입

```
system_name = comanz
user_id = turbo
os_type = Windows
os_version = XP
ip_addr = 163.152.155.237
mac_addr = 00-A0-B0-10-2C-72
patch_number = 2
patch_1 = Q816093
patch_2 = Q282522
program_number = 2
program_list_1 = MS-WORD
program_list_2 = MS-EXCEL
```

이름, 운영체제 종류 및 버전 정보, MAC 주소, IP 주소, 설치된 프로그램의 개수 및 목록, 설치된 패치 개수 및 목록 등의 필드들로 구성된다. <표 3>은 패치 프로파일의 프로토타입이다.

3.3 패치 DB

패치 DB는 서버 DB와 패치 DB로 구성된다. 서버 DB는 클라이언트 관리 프로파일과 클라이언트 환경 정보 프로파일, 그룹 리스트 프로파일을 구성한다. 즉, 서버 DB는 Admin_Tabel, Client_User_Table, Client_Host_Table, Group_Table로 구성된다.

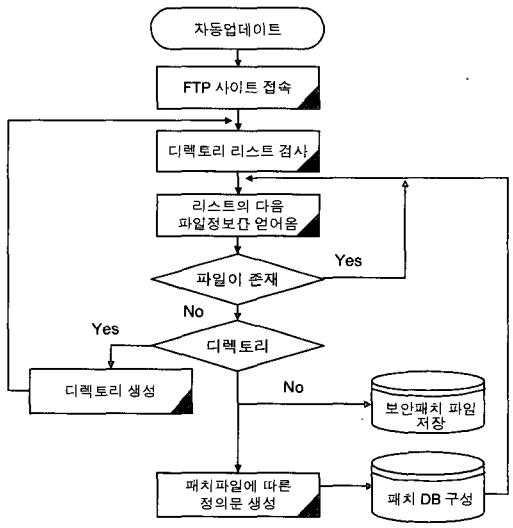
패치 DB는 클라이언트 패치 정보 프로파일과 클라이언트 프로그램 정보 프로파일로 구성된다. 즉, Window_Table, Solaris_Table, Linux_Table로 구성된다.

〈표 4〉 패치 DB 테이블 및 데이터 형식

index	long(unique)
sp_name	varchar(50)
sp_os_version	varchar(50)
sp_file_path	varchar(50)
pre_patch	varchar(50)
is_reboot	bool
checksum	varchar(100)
last_update	date

관리자가 모든 벤더에 매일같이 접속하여 새로운 패치의 존재 여부를 확인하고, 일일이 패치를 다운받아오는 작업을 수행하는 것은 효율성이 떨어진다. 따라서, 본 연구에서는 패치 DB를 자동구성 하는 방안을 연구하였다. Windows와 Soaris의 경우 http 웹 서비스를 통하여 패치를 제공하므로, 자동구성에 어려움이 있어, FTP 사이트를 이용하여 제공하는 Redhat을 대상으로 자동구성 방안을 연구하였다.

FTP를 이용한 자동 검색 및 구성은 벤더들의 FTP 서버에 접속하여 FTP 디렉토리와 패치 파일이름으로 해당 패치파일을 검색하여 패치를 수집하는 방법을 사용한다. FTP 계정정보를 이용해 벤더의 FTP서버에 접속하여 디렉터리의 리스트를 얻어온다. 얻어온 리스트에서 순차적으로 파일존재, 디렉터리/파일 판별의 비교를 거친다. 새로운 디렉터리의 경우 디렉터리를 생성하고 생성된 디렉터리로 이동하여 검색을 진행한다. 새로운 패치 파일이 존재하는 경우 패치 서버에 패치파일을 저장하고 정의문, 체크섬, 날짜, 패치파일의 경로 등으로 패치 DB를 자동으로 구성한다.



(그림 2) FTP 대상의 패치 DB 자동구성 순서도

3.4 패치 분배 및 설치 시나리오

패치 파일을 분배하는 방식에는 크게 Push 방식과 Pull 방식이 있다. Push 방식은 보안패치 서버에서 클라이언트 에이전트에게 패치파일을 밀어주는 방식이다. Pull 방식은 클라이언트 에이전트에서 패치 서버에 있는 패치파일을 당기는 방식이다.

패치 분배 및 설치 시나리오는 패치 분배 초기화, 패치 분배 서비스 요청, 새로운 패치의 등

록으로 3가지의 경우가 존재한다. 앞서 살펴본 보안패치 파일의 분배 방식을 이용하여 각각의 시나리오에 따라 패치를 자동으로 분배하고 설치한다.

3.4.1 패치 분배 초기화 시나리오

① 클라이언트 시스템이 켜지고 패치 에이전트 프로그램이 자동실행(패치 서버에 접속)
② 패치 DB와 클라이언트의 프로파일을 비교하여 설치되어 있지 않은 패치파일의 존재성 확인
③-1 패치파일이 전부 설치되어 있는 경우 : 시나리오 작업 종료
③-2 설치되지 않은 패치파일이 존재하는 경우 : 해당 패치파일을 분배
④ 분배 받은 패치파일을 자동으로 설치
⑤ 클라이언트의 프로파일 정보 업데이트 및 패치 서버로 전송

3.4.2 새로운 패치의 등록 시 시나리오

① 벤더로부터 새로운 패치를 다운 받음
② 패치 서버 관리자 - 새롭게 제작된 패치파일의 안정성 테스트
③ 안정성에 이상이 없는 경우, 새로운 패치파일을 패치 DB에 등록
④ 새롭게 추가된 패치파일을 분배받을 필요가 있는 클라이언트 목록 작성
⑤ 클라이언트 목록에 존재하는 각각의 클라이언트들에게 패치파일 분배
⑥ 분배 받은 패치파일을 자동으로 설치
⑦ 클라이언트의 프로파일 정보 업데이트 및 패치 서버로 전송

3.4.3 패치 분배 서비스 요청 시 시나리오

패치를 분배 및 설치하고자 할 때 사용자가 중요한 문서작업을 하고 있는 경우에는 패치 설치를 거부할 수 있다. 향후 사용자가 패치 서버 매니저에 접속하여 새로운 패치를 분배 받거나 시간대별로 동작하는 스케줄링 기능을 이용하여 패치를 분배 및 설치 할 수 있다.

(1) 클라이언트 에이전트를 이용한 요청에 의한 분배 및 설치 시나리오
① 클라이언트 에이전트에 접속
② 자신의 시스템에 설치되지 않은 패치 목록 확인
③ 목록에서 패치 파일을 선택하여 다운받아서 설치
④ 클라이언트의 프로파일 정보 업데이트 및 패치 서버로 전송

(2) 스케쥴링에 의한 시나리오
① 클라이언트 에이전트의 실행과 동시에 스케쥴링 주기용 타이머 작동
② 타이머 주기에 따라서 정기적으로 패치 서버에 접속
③ 3.4.1. 패치 분배 초기화 시나리오의 ②부터 ⑤번 시나리오까지 수행

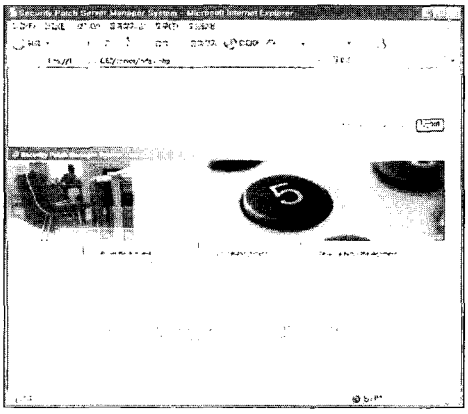
4. 시스템 구현 및 시험

본 연구에서는 JAVA를 이용한 개발을 통하여 다양한 운영체제를 동시에 지원할 수 있도록 하였으며, 객체 지향적 설계 및 구현을 통하여 다른 보안제품에 패치관리시스템을 추가하여 병렬적인 수행이 가능하도록 하였다. 또한, 패치분배 서버에 대한 관리를 Web GUI 환경을 이용하여 관리를 수행할 수 있도록 개발하여 관리자가 편리하게 패치에 대한 추가 등의 작업을 수행할 수 있도록 개발하였다.

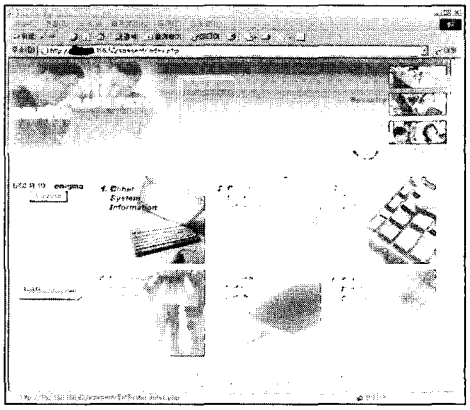
연구 내용에 대한 시험은 개발된 패치분배 서버를 시험망에 설치하고, 여러 대의 이기종의 시스템들을 클라이언트로 설정하여 에이전트 프로

그램을 설치하여 패치서버로부터 자동적으로 패치를 분배받고, 설치하는지의 여부를 시험하였다.

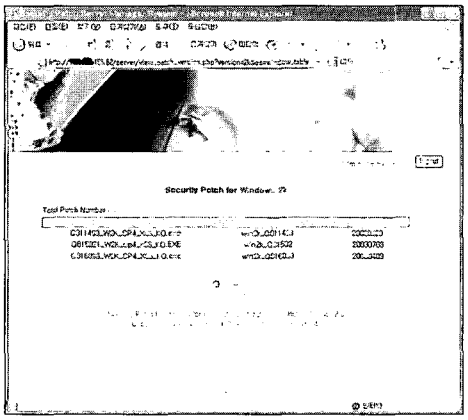
시험결과 패치서버 프로그램, 패치 서버 매니저, 패치 클라이언트 에이전트, 패치 클라이언트 매니저 프로그램들이 정상작동 하는 것을 확인하였고, 패치 서버 매니저에 새로운 패치를 추가하였을 경우 자동적으로 해당 패치를 필요로 하는 클라이언트들을 선별하여 패치를 분배하고 설치하는 기능을 확인하였다. (그림 3)에서 (그림 9)는 각각의 프로그램들이 정상적으로 작동하고, 패치가 자동으로 분배 및 설치되고, 관련 프로파일이 생성 및 갱신되는 것을 확인한 그림들이다.



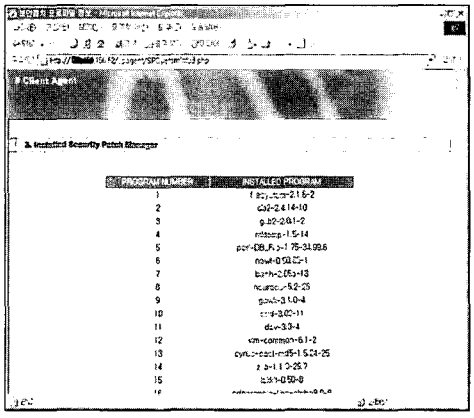
(그림 3) 패치 서버 매니저



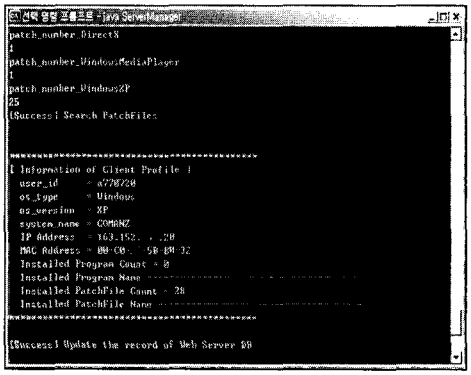
(그림 4) 패치 클라이언트 매니저



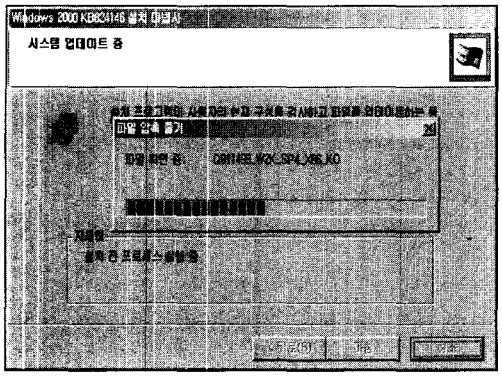
(그림 5) 패치 DB 내의 패치정보



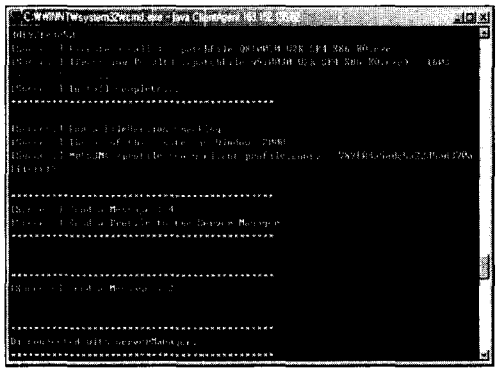
(그림 6) 클라이언트에 설치된 프로그램 정보 확인(Linux)



(그림 7) 클라이언트 프로파일 생성(Windows)



(그림 8) 클라이언트에 패치 설치(Windows)



(그림 9) 패치 서버 프로그램에 프로파일 전송

5. 결론 및 향후 연구방향

본 논문에서는 시스템 보안성 향상을 위한 패치관리시스템을 설계 및 구현하였다. 제안하는 시스템은 대규모 네트워크 내의 클라이언트들에 대한 패치 관리를 자동으로 수행 가능하다. 패치에 대한 자동관리를 통하여 모든 클라이언트 시스템들이 항상 최신의 패치가 설치된 최상의 상태를 유지할 수 있도록 하여 시스템보안 측면에서 매우 효과적인 시스템이다.

향후 연구로는 본 시스템을 대규모 네트워크 및 사설 네트워크 등을 사용하는 다양한 환경에 대한 지원을 효과적으로 수행 할 수 있도록 시스템의 기능 향상 및 안정화를 수행하는 것이다.

참 고 문 헌

[1] CERT/Coordination-Center : CERT/CC statistics, http://www.cert.org/stats/cert_stats.

[2] Bashar, M. A., Krishnan, G., Kuhn, M. G., Low-threatsecuritypatchesandtools, In : Proceedings of the International Conference on Software Maintenance, pp.306-313, 1997.

[3] Shon, T. S., Moon, J. S., Seo, J. T., Im, E. K., Lee, C. W., Safe Patch Distribution Architecture in Intranet Environments//SAM, 2003.

[4] Lee, C. W., Im, E. G., Seo, J. T., Moon, J.S., Kim, D.K., Design of a secure and consolidated patch distribution architecture//Proceedings of the International Conference on Information Networking, 2003.

[5] LLNL, SafePatch, Lawrence Liwrence Livermore National Laboratory.

[6] Comerfore White., ABYSS : A trusted architecture for software protection. In Proc. 1987 IEEE Symposium on Security and Privacy, Oakland, California, IEEE Computer Society Press, pp.38-5, April 1987.

[7] PatchLink, http://patchlink.com.

[8] BigFix Patch Manager, http://www.bigfix.com.

[9] Shavlik HFNetChkPro Security Patch Management, http://www.shavlik.com.

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

[11] MCAFEE, http://www.debian.org/distrib/packages.

[12] Microsoft, http://windowsupdate.micorsoft.com/.

[13] Sun Micorsystems, http://sunsolve.sun.com/.

[14] symantec, http://www.symantec.com/region/kr/press/kr_010507.htm.

서 정 택

1999년 충주대학교 컴퓨터공학과(공학사)

2001년 아주대학교 컴퓨터공학과(공학석사)

2000년~현재 국가보안기술연구소 정보보증연구부 선임연구원

윤 주 범

1999년 고려대학교 컴퓨터학과(이학학사)

2001년 서울대학교 컴퓨터공학과(공학석사)

2001년~현재 국가보안기술연구소 정보보증연구부 연구원

최 대 식

1997년 강원대학교 전자계산학과(학사)

1999년 강원대학교 전자계산학과(석사)

2000년~현재 국가보안기술연구소 정보보증연구부 연구원

박 응 기

1986년 중앙대학교 전자계산학과(학사)

1988년 중앙대학교 전자계산학과(석사)

1988년~1999년 한국전자통신연구원 선임연구원

2000년~현재 국가보안기술연구소 정보보증연구부 책임연구원(팀장)

박 춘 식

1995년 일본 동경공업대학교 전기전자공학과(공학박사)

1989년~1990년 일본 동경공업대학교 초빙연구원

1982년~현재 한국전자통신연구원 부설 국가보안기술연구소 책임연구원

2003년~현재 고려대학교 정보보호대학원 겸임교수