동적 부하균형을 지원하는 서버용 안티바이러스 소프트웨어 설계 및 구현*

최주영** · 성지연** · 방혜미** · 최은정*** · 김명주****

요 약

악성 코드로 인한 피해에 적극 대응하기 위해서는 클라이언트가 아닌 서버 측에서 실행되는 안티바이러스 소프트웨어가 필요하다. 그러나 서버용 안티바이러스 소프트웨어로 인하여 서버의 부하가 가중되는 것은 바람직하지 않다. 본 논문에서는 모니터/에이전트 구조로 멀티프로세서 환경의 서버 시스템에서 수행되는 안티바이러스 소프트웨어를 개발하였다. 이 소프트웨어는 안티바이러스 엔진의 주요 특징을 반영하여 서버에서 동적 부하 균형을 지원해줌으로써 효율적인 수행 환경을 제공한다. 악성코드 검색율과 검색 속도에 대한 성능 측정 결과는 서버용 안티바이러스 소프트웨어로서의 장점과 특징을 입증해준다.

Design and Implementation of Anti-virus Software for Server Systems supporting Dynamic Load Balancing*

Ju-Young Choi** · Ji yeon Sung** · Hemi Bang**
Eun-Jung Choi*** · Myuhng-Joo Kim****

ABSTRACT

It is more desirable to execute AV software on server systems rather than on clients for the minimization of damages due to malicious codes. AV software on server system, however, may aggravate the load of server system. In this paper, we propose a new AV software executed on server system without additional loads with a monitor and multi-agent model. The new AV software supports dynamic load balancing that reflects the features of AV engine, and thus it can be executed efficiently on server systems. The results of performance evaluation on the AV software attest to the strong points of the new AV software.

Key words : Anti-Virus, Server Security, Multiprocessor System, Load Balancing

---

* 본 연구는 2006년도 서울여자대학교 교내 학술연구비 지원에 의하여 수행되었습니다.
** 서울여자대학교 대학원 컴퓨터학과
*** 서울여자대학교 정보통신교육원
**** 서울여자대학교 정보미디어대학 정보보호학 전공

1. 서 론

정보화 사회의 기반인 인터넷 환경에서 침해사고는 유명 웹 사이트의 경우 웜과 바이러스 제작자들에게 아주 매력적인 전파수단이 될 것으로 예상하고 있다[1].

서버는 광범위한 네트워크 상에서 서버와 클라이언트 간에 데이터 송수신이 일어난다. 도메인 내부적으로 많은 클라이언트들이 연결되어 클라이언트의 전자우편을 일차적으로 수신, 전달한다. 그리고 홈페이지 서버로 운영되거나 클라이언트 파일 송수신에 이용하기도 한다.

이처럼 중요한 위치를 차지한 서버에 동적 부하균형(load balancing)을 지원하는 안티바이러스(AV, Anitvirus) 엔진을 장착한다. 다중 CPU의 장점을 가진 서버에 적용함으로 하나의 프로세서에 과중한 부하를 감소시키며 선택적인 멀티프로세서는 검사 및 치료의 효율성을 높이고자 한다.

본 논문은 기존 국내외 서버용 AV 엔진을 특징을 비교 분석하였고, AV 엔진의 구조를 연구하였다. 실제적인 바이러스 검사 및 치료하는 AV 에이전트, CPU 부하 측정과 할당을 관여하는 AV 모니터, 바이러스 종류별로 분류된 바이러스 DB, 원격에서 검사 로그를 확인 할 수 있는 Web Log Report, 실시간 네트워크 패킷 검사, 최신 바이러스를 업데이트하는 Smart Update를 제안하고자 한다.

2. 관련 연구

2.1 국내외 기술 개발 현황

다중 CPU를 이용한 병렬처리 서버용 AV 엔진을 설계 및 구현에 앞서, 국내외 백신 업체들의 제품을 <표 1>에서 분류하였다.

〈표 1〉 국내외 서버용 AV 엔진 현황

분류	국외		국내		
	S업체	T업체	A업체	H업체	T업체
윈도우 서버용	Antivirus Command Line Scanner	Server Protect for Microsoft Windows/ Novell Netware	V3Net for Windows Server SE	바이로봇 Advanced Server	터보 백신 Windows Server
유닉스 /리눅스 서버용	Antivirus Command Line Scanner	Server Protect for Linux	V3 Net Scan2001, V3 Viruswall Filescan	바이로봇 Unix /Linux Server	-
다중CPU 기능여부	x	x	x	x	x

국내 회사에서 개발하여 판매하고 있는 서버용 AV 엔진은 A업체의 V3 Net Scan 2001과 V3 Viruswall Filescan[2], H업체의 바이로봇 Unix/Linux Server[3], T업체에서는 윈도우 서버용만 지원하고 있다[4]. 국외 업체의 서버용 AV 엔진은 S업체의 Antivirus Command Line Scanner[5], T업체의 Server Protect for Linux[6]가 있다.

현재 상용화되고 있는 서버용 AV 엔진의 기본 기능은 실시간 감시 기능, 바이러스 패턴 업데이트 기능, 다양한 압축 파일 검사 기능, 사용자 사용이 편한 인터페이스 제공 등이다. 하지만, 멀티프로세서가 장착된 서버의 특성을 살려 다중 CPU를 제어하거나 부하를 측정하는 등의 기능은 제공되지 않음을 알 수 있다.

2.2 AV 엔진 분석

AV 엔진의 핵심 기술은 검색 알고리즘이다. 시스템을 바이러스에 감염시키는 바이러스 감염 실행 패턴(바이러스 시그니쳐)을 검색 대상 파일에서 찾아내는 방법이다.

〈표 2〉 매칭 알고리즘 분류

기능 / 종류	설 명	특 징
Brute-Force	길이가 n인 텍스트(T)의 모든 위치에서 길이가 m인 패턴(P)가 시작되는지 하나씩 맞춰보는 방법	• 이해용이 • 최악의 경우 O(mn)
KMP	문자를 비교하는 과정에서 일치하지 않는 경우가 발생하면 다시 비교할 위치를 결정할 때 이미 비교한 패턴에 있는 정보를 활용하여 문서의 문자 위치를 나타내는 변수i를 결정	• 반복되는 문자들이 많은 스트링에서 반복되는 문자들이 많은 패턴을 찾을 경우 유리 • 최악의 경우 O(m+n)
Boyer-Moore	KMP알고리즘과 유사한 방식이지만 패턴과 문서를 비교할 때 오른쪽에서 왼쪽으로 비교하는 것으로, 전처리 단계에서는 문장에 나타날 수 있는 각 패턴에 대해 불일치가 발생할 경우 SKIP정도를 결정	• bad character method와 good suffix method 중에서 패턴을 오른쪽으로 더 많이 이동하는 것을 선택하여 사용 • 최악의 경우 O(mn)

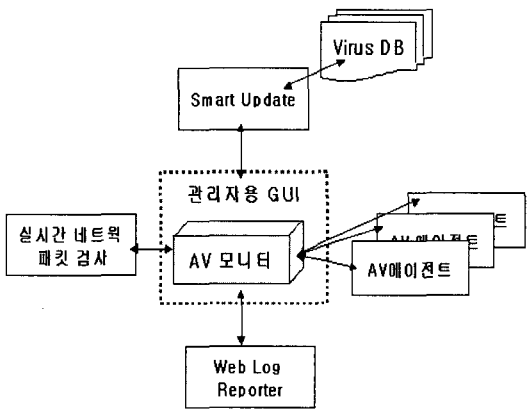
AV 엔진의 성능을 향상시키기 위한 방법으로는 크게 두 가지가 있는데, 첫째는, 검사 속도를 빠르게 하는 것이고 둘째는, 진단율을 높이는 것인데, 진단율을 높이는 것의 경우는 바이러스 시그니처(Signature) DB를 얼마나 잘 구성하느냐 혹은 다형성 바이러스와 같은 변형 바이러스를 잘 진단하는 기법을 개발하느냐에 있다고 볼 수 있다.

일반적으로 패턴 매칭(pattern matching)은 문자열 중 지정된 서브 문자열을 찾는 것으로 서브 문자열이 있으면 참이 되고, 없으면 거짓이 되는 연산을 말한다. 위의 <표 2>에서 기존 매칭 알고리즘을 설명과 특징이다.

병렬처리 기능을 높이기 위해 가장 중요한 속도 향상을 위해 사용될 수 있는 매칭 알고리즘[7-9]을 연구하고 본 논문에서 구현한 다중 CPU 기반의 병렬처리 AV엔진은 반복되는 문자열이 많은 패턴에 적합한 KMP 알고리즘을 적용하였다.

3. 병렬처리 서버용 AV 엔진 설계

기존 안티바이러스 엔진은 다중 CPU 기능을 갖는 서버에 설치되어 동작하더라도 다중 CPU를 관리자가 활용할 수 없었다. 따라서 빠른 탐지를 요하는 안티바이러스 엔진의 속도를 저하시키고 현재 서버의 고유 업무를 행하는 다른 프로세스의 활동에도 영향을 미치는 문제점이 있다. 또한 바이러스도 나름대로 유행과 세대적 흐름을 가지고 있는데 이런 컴퓨터 바이러스의 특징에 상관없이 검사대상(바이러스 DB)을 통합하여 검사함으로써 컴퓨터 바이러스로부터 서버를 효율적으로 보호하기 어렵다.



(그림 1) 다중 CPU의 AV 엔진 설계

이러한 문제점을 해결하기 위하여 병렬처리 서버용 AV 엔진을 다음과 같이 다중 CPU를 활용

하여 AV 엔진의 속도 증가와 바이러스 DB에 카테고리를 적용하여 검사 항목을 선택할 수 있는 기능을 설계한다. 위에 (그림 1)은 다중 CPU의 AV 엔진 설계이다.

병렬처리 서버용 AV 엔진은 바이러스 감염 여부를 핵사 코드값으로 비교하는 AV 에이전트, AV 엔진의 전체를 관리하는 AV 모니터, 관리자용 GUI로 구성되어진다. 또한 바이러스 검색을 위한 바이러스 DB, 최신 바이러스 정보를 갱신하여 주는 Smart Update, 검색 후의 로그를 남기기 위한 Web Log Report로 구성된다.

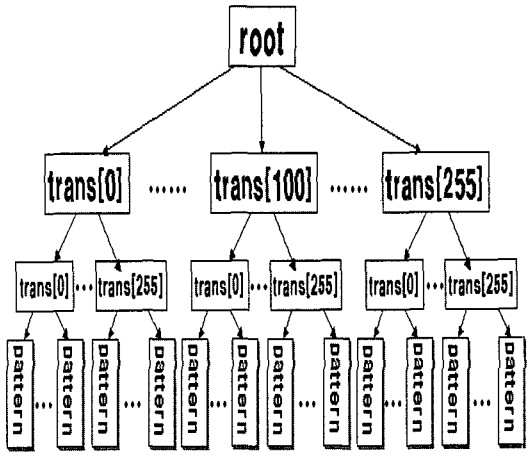
3.1 AV 에이전트(Agent)

AV 에이전트는 안티바이러스 엔진으로 바이러스 유무를 진단하는 기능을 수행한다. AV에이전트는 CPU 1을 포함하여 CPU n개까지 독립적으로 설치된다. 이러한 초기 설치는 모니터링 모듈을 통해서 이루어진다. 병렬처리 서버용 AV에이전트 자체는 클라이언트용 AV 엔진과 유사하다.

AV 에이전트의 기능은 단순히 바이러스 유부를 판단하는 것이지만, 그러한 판단을 위해 사용되는 패턴 매칭 알고리즘을 무엇을 쓰느냐에 따라 성능이 크게 좌우된다. 따라서 AV 에이전트의 바이러스 진단 속도를 높이기 위해 트리 구조와 KMP 알고리즘을 동시에 적용한다.

바이러스 DB를 2Depth를 가지는 트리 구조를 형성한 후 2Depth 이후부터는 KMP 알고리즘을 사용해 매칭하는 방법을 사용하게 된다. 아래의 (그림 2)는 전체적인 트리 구조 형성을 보여주는 것으로 핵사값이 1~256개의 형태로 이루어져 있으므로 1Depth에서 256개의 가지가 생성되고, 2Depth에서 각각의 값 아래에 256개의 가지가 생성되며, 2Depth 아래에는 그냥 나머지 핵사 값으로 이루어진 바이러스 DB 시그니쳐 값들이 연결되어있는 것을 알 수 있다.

바이러스 DB를 트리 구조로 형성해서 검사하게 되면, 바이러스 시그니쳐 각각에 대해 매칭 알고리즘을 적용해 바이러스 DB 파일의 처음 시그니쳐 부터 마지막 시그니쳐까지 모두 매칭하고, 다시 또 매칭하는 반복적인 작업을 하지 않고, 단지 트리를 생성해서 메모리에 올린 후 검사 대상 파일을 핵사화 한 후 트리 구조를 따라가면서 매칭하게 되므로 검사 속도가 줄어들게 된다.

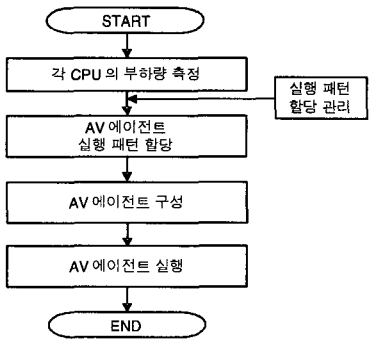


(그림 2) 바이러스 DB 구조

3.2 AV 모니터(Monitor)

AV 에이전트와 바이러스 DB 파일을 관리하며, 검사 실행 환경 설정, 업데이트 모듈의 실행, 패킷 스캔 모듈의 실행 등 통합 관리 시스템이라고 할 수 있다.

AV 모니터는 n개의 CPU를 구비한 병렬처리 시스템의 하나의 CPU에 로드되어 각 CPU의 부하에 따라 AV 에이전트의 실행 패턴을 각 CPU별로 할당하고, 각 CPU에 할당된 실행패턴에 따라 각각 AV 에이전트를 구성하고 이들 각각이 할당된 CPU를 통해 실행되도록 처리함으로써 멀티 CPU의 고유 특성인 병렬처리 기법을 바이러스 검사 및 치료 시에도 이용할 수 있어 바이러스 검사 및 치료 속도를 향상시킬 수 있게 된다. 다음 (그림 3)은 AV 모니터의 흐름도이다.



(그림 3) AV 모니터 흐름도

3.2.1 CPU 부하량 측정

AV 모니터는 병렬처리 시스템의 모든 CPU의 트래픽(각 CPU별로 현재 수행중인 프로세스나 처리중인 명령수 등)을 OS로부터 파악하여 각 CPU의 부하량을 측정한다.

3.2.2 실행패턴 할당 관리

AV 에이전트가 실행되는 각 CPU의 부하량을 모니터링하고, 이 모니터링된 각 CPU별 부하량 정보를 기록하여 AV 모니터 재실행시 실행패턴 할당시 이를 반영하도록 한다.

CPU에 할당된 AV 에이전트를 실행함에 의해 예기치 못한 CPU 부하가 해당 CPU에 발생했다면, 실행 패턴 할당 관리는 AV 모니터 재실행시 해당 CPU에 CPU 부하를 야기시켰던 AV 에이전트 실행을 배제하도록 하는 등의 처리를 통해 보다 효율적인 바이러스 검사 및 치료가 가능하다.

3.2.3 실행패턴 할당

CPU 부하량 측정부에 의해 측정된 각 CPU의 부하량에 따라 각 CPU별로 AV 에이전트의 실행 패턴을 할당한다. CPU가 여러개인 병렬처리 시스템에서 속도향상을 위해서는 수행중인 프로세스수나 처리중인 명령수가 많아 CPU 부하량이 큰 CPU에서는 상대적으로 처리하는 업무가 최소가 되도록하고, 수행중인 프로세스수나 처리중인 명령수가 적어 CPU 부하량이 작은 CPU에서는 상대적으로 처리하는 업무가 최대가 되도록 함으로 처리속도 향상을 가져온다.

3.2.4 AV 에이전트 구성

실행패턴 할당에 의해 할당된 실행 패턴에 따라 각 CPU를 통해 로드되어 실행될 AV 에이전트들을 구성한다.

AV 에이전트를 구성하는 요소들은 바이러스 패턴 DB인 전체(all), 최신(daily), 윈도우 바이러스(Windows), 유닉스/리눅스(Unix, Linux), 웜(Worm), 스크립트(Script)등 바이러스 군에 따라 각각 이들을 검사하고 치료하게 된다.

3.2.5 AV 에이전트 실행

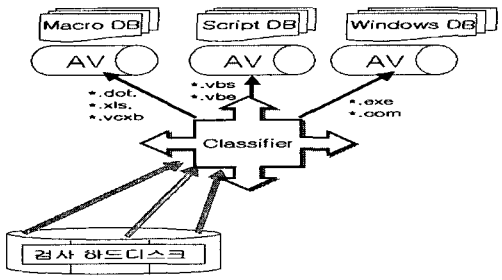
구성된 AV 에이전트들을 각각 할당된 CPU에 로드시켜 실행하도록 OS에 신호하여 각 CPU별 동작을 하는 AV 에이전트들이 실행되도록 한다.

AV 엔진의 편리한 통합 관리를 위하여 GUI 환경을 지원한다. Unix와 Linux에서 사용되어지는 GTK+[10] 프로그래밍 언어를 사용한다.

3.3 바이러스 DB

AV 엔진의 검사 속도 향상을 위해서는 패턴 매칭 알고리즘을 적절하게 사용하는 것도 좋은 방법이지만 또 한 가지 효과적인 바이러스 DB를 구축하는 것이다. 본 논문에서는 파일 분류자(Classifier)를 이용해 파일을 종류별로 분류하여 진단율의 속도를 향상 시킨다.

바이러스에 감염되는 파일을 잘 분석해 보면, 일정한 패턴을 가지는 것을 알 수 있다. 예를 들어 스크립트 바이러스에 감염된 파일의 확장자는 대부분 "*.vbs "이며, 매크로 바이러스에 감염된 파일의 경우 대부분 확장자가 "*.dot, *.xls, *.vcxb "임을 알 수 있다. 다음 (그림 4)는 파일 분류자를 통해 바이러스 DB를 분류한 예를 보여준다.



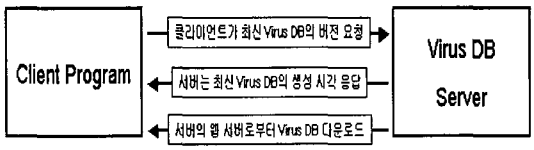
(그림 4) 파일 분류자 사용 예

감염 파일의 확장자 정보와 같은 특성을 이용해 파일 분류자(Classifier)가 AV엔진으로 검사 대상 파일을 보내기 전에 미리 분류하여, 관련 바이러스 DB와 검사를 실행할 수 있도록 해주어 전체 바이러스 DB 파일과 검사하지 않아도 됨으로 검사 시간을 줄이게 된다.

본 논문에서는 공개 안티바이러스 Clam Anti-Virus[11] 엔진에서 사용하는 바이러스 DB에 파일 분류자를 적용하였다.

3.4 Smart Update

AV 엔진에 의한 바이러스 검사시 참조되는 바이러스 정보를 업데이트하여 상기 바이러스 정보 DB를 갱신한다. 다음 (그림 5)는 Update 처리 과정을 보여준다.



(그림 5) Update 처리 과정

AV 모니터에 의해 할당된 CPU별로 실행되는 AV 에이전트은 패턴 매칭을 통해 검사 대상이 바이러스 코드를 포함하고 있는지 검사하게 되는데, 패턴 매칭시 참조되는 바이러스 정보를 Smart Update를 통해 업데이트 서버 등으로부터 수신하여 바이러스 정보 및 DB를 갱신한다. 따라서, 바이러스 정보 업데이트를 통해 신규 바이러스를 검사하고, 치료할 수 있게 된다.

3.5 Web Log Report

병렬처리 서버용 AV 엔진 실행 후 처리 결과에 대한 로그 정보를 웹 페이지를 통해 사용자에게 제공한다. 날짜, 감염된 파일, 감염된 바이러스 이름 등의 정보를 제공한다.

3.6 네트워크 패킷 검사

네트워크를 통해 서버로 들어오는 패킷에 대한 바이러스의 유무를 진단하며 바이러스가 발견 시 로그를 남기고, 콘솔에 경고 창을 띄워 실시간 검사가 가능하도록 한다.

패킷을 바이러스 시그니쳐와 비교한 후 바이러스인지 판별하는 실시간 패킷 검사 구현 과정은 다음과 같다.

(1) hostname을 구한다.
(2) 목적지가 hostname인 패킷을 받고 tcp, ip, icmp인지 검사한다.
(3) 패킷 헤더를 검사하여 destination port number와 헤더를 제외한 data길이를 측정한다.
(4) virus_db에 있는 virus 패턴을 buffer로 하나씩 읽어 들어와서 port와 signature를 구분자(= :)를 이용하여 분류하고 받아들여진 패킷의 destination port와 data길이가 각각 일치한다면 일대일 매칭을 시켜보며 바이러스인지 검사한다.
(5) (4)번으로 돌아가 virus_db에 있는 모든 signature를 검사한다. 단 바이러스 발견시 그 패킷은 더 이상 virus_db에 있는 다른 signature 들과 매칭하지 않고 바이러스를 알려주는 메시지 창을 띄워주며 로그에 "시간 : 포트 : 바이러스이름 : 공격ip 등의 순서로 남긴다.

4. 병렬처리 서버용 AV 엔진 구현

C언어를 이용하여 구현한 Unix/Linux 환경에서 수행되는 병렬처리 서버용 AV 엔진이다.

4.1 구현 환경

- 플랫폼 : i386 기반의 GNU/Linux, SunSolaris
- 라이브러리 : libpcap, libclamav(Clam AV scnning library)
- 쓰레드 : Pthread
- 웹 서버 : Apache 1.3.23
- 프로그래밍 언어 : PHP 4.2.3, XML
- 데이터베이스 : Mysql 3.23.53
- GUI 인터페이스 : GTK+

4.2 병렬처리 서버용 AV 엔진 기능

(1) AV 엔진의 활성화 기능
병렬처리 서버용 AV엔진의 수행 여부를 판별할 수 있는 것으로 사용자의 요구에 따라 바이러스 검사를 실행 또는 중지 할 수 있다.

(2) 쓰레드(CPU) 개수 설정 기능
바이러스 검사에 사용되는 쓰레드 개수를 설정한다. 실행 AV 엔진에서 사용할 쓰레드(=CPU)의 개수를 정의한다.

(3) 바이러스 발견 후 처리 기능
- 바이러스가 발견된 파일을 삭제
- 바이러스가 발견된 파일을 특정 디렉토리로 옮김
- 아무 일도 수행하지 않음

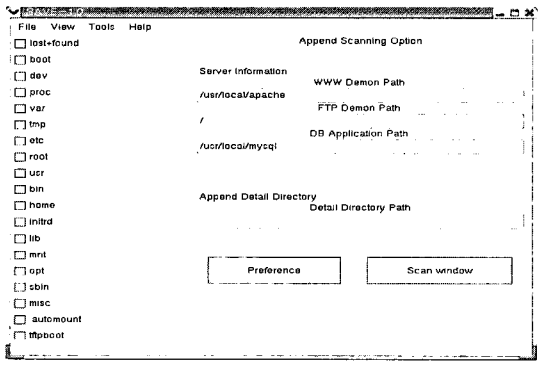
(4) 바이러스 시그니쳐 DB 선택 기능
서버의 파일들과 패턴 매칭을 하게 될 바이러스 시그니쳐 DB는 다음 6가지로 선택적으로 중복 설정 할 수 있다.
- 최신 바이러스
- 윈도우즈 바이러스
- 유닉스/리눅스 바이러스
- 웜 바이러스
- 스크립트 바이러스
- 전부

(5) 멀티쓰레드 기능
Pthread를 이용하여 파일 검색 단위로 멀티쓰레드 작업을 수행한다.

(6) 압축파일 검사 기능
다양한 압축파일 “zip”, “rar”, “arj”, “zoo”, “jar”, “lzh”, “tar”, “deb”, “tar.gz”, “tgz”등을 검사한다.

4.3 병렬처리 서버용 AV 엔진 인터페이스

시작 메인 화면은 크게 두 부분으로 나뉘며, 메뉴 선택을 할 수 있는 메뉴바 부분과 검사 디렉토리 설정 부분으로 이루어져 있다. 다음 (그림 6)은 AV 엔진의 메인 화면이다.



(그림 6) AV 엔진의 메인 화면

메뉴 구성은 크게 4가지(File, View, Tools, Help)로 되어 있다. [File] 메뉴는 검사실행 및 환경설정에 관련된 선택 메뉴로 구성되어 있고, [View] 메뉴는 바이러스 DB 정보 및 검사 로그, 업데이트 로그를 보여주도록 하는 선택 메뉴이다. [Tools] 메뉴는 스마트 업데이트 실행에 관한 선택 메뉴이며, 마지막으로 [Help] 메뉴는 AV 엔진의 매뉴얼과 정보를 보여주는 선택 메뉴이다.

5. 병렬처리 서버용 AV 실행 시나리오 및 테스트

구현된 서버용 AV 엔진을 시나리오에 적용하여 검색율과 검색 속도에 대한 성능을 테스트한 결과를 보여준다.

5.1 실행 시나리오

서버용 AV 실행 시나리오 가정은 다음과 같다.

- 서버의 사용 용도 : Web server
- 서버에 장착된 CPU 수 : 4개
- 감염 파일 처리 방법 : 삭제(remove)
- 패킷 검사 기능 사용 여부 : 사용함 (Use)
- 검사 디렉토리 : /home, /usr/local
- 바이러스 시그니쳐 DB : 최근 바이러스DB (Current viruses)

서버의 사양 정보는 다음 <표 3>과 같다.

〈표 3〉 테스트 서버 사양 정보

Compaq linux	Sun blade 2000
CPU : Intel(R) Pentium(R) IIICPU 1400MHz (cache : 512Kb)-2개	CPU : UltraSPARC-III+ 900MHz(cache : 8MB)-2개
Memory : 512Mb	Memory : 2GB
OS : RedHat 7.3	OS : Sun Solaris
Kernel version : 2.4.18-3smp	Kernel version : 5.8

〔단계 1〕 Smart Update 실행

검사를 시작하기 전에 검사에 사용되는 바이러스 DB의 최신 정보를 유지해야 할 필요가 있으므로, Smart Update를 실행한다.

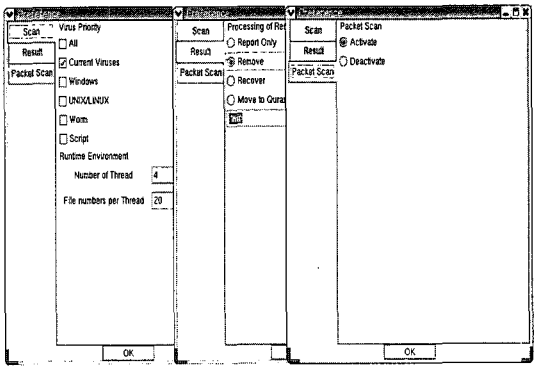
〔단계 2〕 실행 환경 설정

실행환경 설정을 위해 환경설정 창을 실행시키기 위해 [File]메뉴의 하위메뉴인 [Preference]를 선택한다(그림 7).

(1) **Scan 설정** : 최근 바이러스 DB에 대해 검사하고자 함으로 바이러스 DB 종류는 [Current viruses]로 선택, 서버에 장착된 CPU가 4개이므로 생성 쓰레드 수를 4로 설정, 쓰레드별 처리 파일 수는 20개로 설정

(2) **Result 설정** : 감염된 파일에 대한 처리는 삭제하는 것으로 가정하였으므로, [Remove]로 설정

(3) **Packet Scan 설정** : 패킷 검사를 사용하는 것으로 가정하였으므로, 기본 값으로 설정



〈그림 7〉 시나리오 〔단계 2〕 - 환경설정

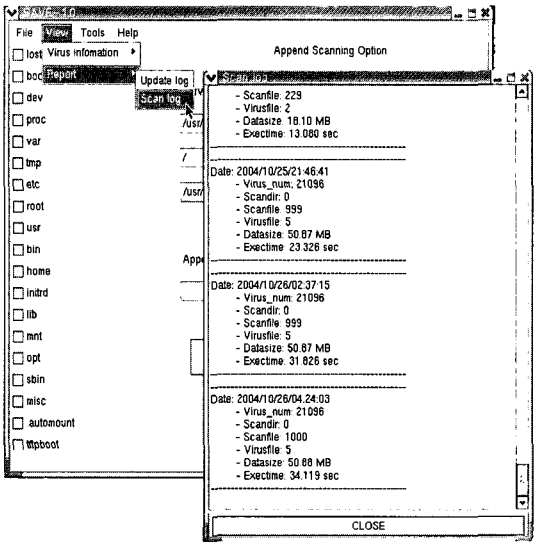
〔단계 3〕 검사 디렉토리 설정

서버의 사용 용도가 웹 서버이므로 웹서버의 주요 사용 디렉토리인 '/usr/local/apache'로 설정한다. 그리고 검사 하고자 하는 디렉토리가 '/home'과 '/usr/local'로 가정하였으므로, '/home'은 루트('/') 아래의 주요 디렉토리를 선택하는 부분에서 설정한다. '/usr/local'의 경우는 세부 디렉토리 설정 창에 직접 입력한다.

〔단계 4〕 검사 실행 및 검사 로그 확인

[File] 메뉴-[Scan window] 메뉴를 선택하거나, 메인 화면 우측 하단의 'Scan window' 버튼을 클릭하게 되면 검사 실행 창을 통해 검사 결과를 눈으로 확인할 수 있다.

검사 결과는 검사결과는 (그림 8)와 같이 [View] 메뉴-[Report]-[Scan log]메뉴를 통해 재확인한다.



(그림 8) 시나리오 〔단계 4〕 - 로그 확인

5.2 테스트 결과

실행 시나리오에 따른 테스트 결과 바이러스 검색율과 바이러스 검색 속도는 다음과 같다.

5.2.1 바이러스 검색율

국내외 상용 안티바이러스 엔진과 서버용 AV 엔진과의 검색율을 벤치마킹한 것이다<표 4>.

〈표 4〉 검색율 벤치마킹

AV 엔진 종류 / 바이러스 종류	A사	H사	서버용 AV
Windows virus (500개)	211	52	476
Unix/Linux virus (50개)	18	23	44
Worm (100개)	72	85	95
Script (100개)	56	67	92
총 750개 (100%)	357 (47.6%)	227 (30.2%)	707 (94.3%)

위의 표에서 보듯이, 국내 안티바이러스 개발 업체인 A사, H사의 상용 안티바이럿 엔진 제품에 비해 구현된 서버용 AV의 검색율이 높게 나왔음을 알 수 있다. 또한 Unix/Linux 바이러스의 검색율도 다른 두 상용 안티바이러스 엔진 제품에 비해 두배 정도 높다는 점에서, 서버용 AV엔진은 Unix/Linux 서버 환경에 적합하다고 볼 수 있다.

5.2.2 바이러스 검색 속도

서버용 AV 엔진에 멀티쓰레드 기법을 사용하여 쓰레드를 생성하여 실행하였을 때와, 그렇지 않았을 때를 비교하여 속도면에서 어떠한 성능의 차이 비교하였다. 다음 <표 5>, <표 6>은 검색속도에 대한 결과표이다.

(1) Compaq linux

2 CPU server, 구현된 서버용 AV 엔진만 수행

〈표 5〉 Compaq linux 검색속도 결과표

실행차수 / Thread 수	1차 (sec)	2차 (sec)	3차 (sec)
1	39.183	39.556	38.333
2	23.726	24.303	24.461
4	23.848	23.986	23.948
16	23.621	23.700	23.675
256	23.444	23.847	23.837

(2) Sun blade 2000

2 CPU server, Web deamon, DBMS 수행

〈표 6〉 Sunblade2000 검색속도 결과표

실행차수 / Tread 수	1차 (sec)	2차 (sec)	3차 (sec)
1	105.943	254.353	133.027
2	72.748	97.936	84.057
4	75.906	76.446	80.908
16	74.810	78.116	79.600
256	72.596	74.428	81.426

검색 속도 테스트에 사용된 두 대의 서버는 모두 2개의 CPU가 장착된 서버로 각 서버의 시스템 사양 및 사용 용도에 따른 실행 프로세스 수의 차이에 따라 속도의 차이가 있다. 위의 결과에서 살펴 볼 때, 쓰레드를 생성하지 않았을 경우와 생성한 경우의 속도는 두 대의 서버에서 동일하게 많은 차이가 난다는 것을 확인 할 수 있다. 또한 생성 쓰레드 수는 서버에 장착된 CPU의 수가 일치할 경우 속도의 향상이 있음을 보여준다.

6. 결 론

본 논문에서는 악성 에이전트들에 의해 감염된 파일을 검사할 수 있는 안티바이러스 엔진에 대해 연구하였으며, 서버의 특징인 다중 CPU가 장착된 Unix/Linux 서버에서의 효율적인 안티바이러스 엔진의 실행에 초점을 맞추어 설계 및 구현하였다. 다중 CPU의 성능을 활용하기 위하여 CPU 부하 측정 및 동적 부하균형 기법은 Pthread 라이브러리를 이용하였다. 바이러스 DB 파일은 파일 분류자를 통해 바이러스의 종류별로 분류하여 관리자에게 선택기능을 제공하였다. 이러한 분류는 최신 바이러스 및 서버 용도에 따라 빈번히 발생하는 바이러스 종류에 대한 선택적으로 검사토록 하여 효율적인 검사 실행을 한다. 또한, Unix/Linux 환경에서 사용자 편의성을 높이기 위해 GTK+ 프로그램을 사용하여 그래픽 인터페이스를 구현하였다. 서버용 실행 시나리오를 통하여 멀티프로세서 적용 여부에 따른 검색율과 검색 속도에 대한 테스트를 하였다.

다중 CPU 서버에서의 안티바이러스 엔진에 대한 연구는 초기 단계이지만, 하드웨어의 기술 발달은 이 분야에 대한 연구의 필요성을 보여준다. 따라서, 멀티프로세서 성능 향상에 대한 연구 및 안티바이러스 엔진의 검사의 정확성과 검사 속도에 대한 성능 향상에 관한 연구가 필요하다.

참 고 문 헌

[1] 한국정보보호진흥원, "2005년 12월 인터넷 침해사고 동향 및 분석 월보", 2005년 연말 특집'.
[2] 안철수 연구소, http://home.ahnlab.com
[3] 하우리, http://www.hauri.co.kr
[4] 터보백신, http://www.everyzone.com
[5] Symantec, http://www.symantec.com
[6] Trendmicro, http://www.trendmicro.co.kr
[7] Brute-Force algorithm, http://www-igm.univ-mlv.fr/~lecroq/string/node3.html#SECTION0030
[8] KMP algorithm, http://www-igm.univ-mlv.fr/~lecroq/string/node8.html #SECTION0080
[9] Boyer-Moore algorithm, http://www-igm.univ-mlv.fr/~lecroq/string/node14.html#SECTION00140
[10] GTK+, http://www.gtk.org
[11] Clam AntiVirus, http://www.clamav.net

최 주 영
1999년 서울여자대학교 컴퓨터학과(이학사)
2003년 서울여자대학교 컴퓨터학과(이학석사)
2006년~현재 서울여자대학교 컴퓨터학과 박사과정

성 지 연
2002년 서울여자대학교 컴퓨터공학과(공학사)
2002년~2004년 (주)ODNK 연구원
2005년~현재 서울여자대학교 컴퓨터학과 석사과정

방 혜 미

2005년 서울여자대학교
컴퓨터공학과(공학사)
2005년~현재 서울여자대학교
컴퓨터학과 석사과정

최 은 정

1997년 서울여자대학교
전산과학과학과(이학사)
2000년 서울여자대학교
컴퓨터학과(이학석사)
2005년 서울여자대학교 대학원
컴퓨터전공(이학박사)

김 명 주

1986년 서울대학교
컴퓨터공학과(공학사)
1988년 서울대학교
컴퓨터공학과(공학석사)
1993년 서울대학교
컴퓨터공학과(공학박사)
1993년~1995년 컴퓨터신기술 공동연구소
특별연구원
2003년~2004년 미국 펜실바니아대학교(UPen)
객원 연구원
1995년~현재 서울여자대학교 정보보호학전공 교수