IPS 구축을 위한 Embedded Linux 성능 비교

양중규*, 류진영*, 남진우*, 장승주*
동의대학교 컴퓨터 공학과
e-mail : ros207@naver.com, dmdeker@naver.com, yang8kyu@hotmail.com, sjjang@deu.ac.kr

Performance Comparison of Embedded Linux for IPS

Seung-Ju Jang*, Jin-Young Ryu*, Jin-Woo Nam*, Joong-kyu Yang*
*Dept. of Computer Engineering, Dongeui University

요 약

본 논문은 IPS 구축에 있어서 핵심적인 역할을 수행하는 운영체제를 위해 최근 각광을 받고 있는 Embedded Linux Kernel 의 성능을 테스트하기 위한 환경을 구축한다. Benchmarking tool 인 UnixBench 프로그램을 사용하여 성능을 테스트한다. 또한 standard kernel 의 성능 테스트도 같이 포함하여 종합적으로 성능 결과를 비교 분석한다. 이런 실험 결과를 통해서 IPS 시스템에 적합한 임베디드 리눅스 운영체제 결정을 하고자 한다.

1. 서론

현재 국 내외 적으로 네트워크 시스템은 질적으로나 양적으로 매우 빠른 발전을 보이고 있고, 이러한 네트워크에 대한 바이러스 및 내, 외부 사용자에 의한 고의적인 시스템 침입 등을 막기 위해 종합적인 보안 시스템도 연구되어왔다. 하지만 현재의 네트워크망은 그 범위가 아주 넓고 또한 매우 복잡해지고 있다. 이에 따라 보안 취약점이 많이 생겨났고 이를 이용한 다양한 해킹들이 지속적으로 발생하고 있다. 특히 광 대역 통합 망과 같은 차세대 네트워크 시스템이 구축되어진다면, 유, 무선 망의 통합에 따른 새로운 인터넷 침해 등이 발생하고, 이에 따른 새로운 정보보호 시스템이 필요하게 되는데, 이 같은 보안 허점을 막아주는 솔루션으로 최근 침입방지시스템(IPS)이 많은 주목을 받고 있다.

IPS 는 기존의 방화벽(Firewall)과 침입탐지시스템(IDS)이 가지고 있는 기능 뿐만 아니라, 복잡해진 네트워크 망 및 새로운 IT 서비스에 따른 새로운 공격 형태를 분석하고 결과를 통합, 그 결과가 네트워크 인프라 및 서비스에 주는 영향을 실시간으로 분석하는 기술 등이 포함 된다. 이 같은 새로운 보안 시스템을 구축하기 위해서는 무엇보다도 이런 시스템을 관리 및 운영할 운영체제의 선정이 매우 중요하다고 볼 수 있다. 특히 네트워크 관리 및 서비스, 그리고 보안 시스템 관리라는 특정 목적을 위한 운영체제로서 Embedded Linux 가 비용 및 Customizing 등을 감안해 매우 좋은 대안이 될 수 있다.

본 논문에서는 먼저 IPS 구축을 위한 성능 테스트에 사용된 Embedded Linux 들의 특징에 대하여 각각 알아보고, 성능 측정을 위한 환경을 구축한다. 다음으로, benchmarking 프로그램을 이용한 성능 측정 결과를 비교 분석 한 후, 결론을 내린다.

2. 관련연구

2.1 IPS

침입 방지 시스템 (IPS : Intrusion Prevention System)은 공격에 대한 탐지만을 하는 침입 탐지 시스템의 한계성을 뛰어넘는 보안시스템으로써, 공격 Signature 를 찾아내 네트워크에 연결된 기기에서 수상한 활동이 이뤄지는지를 감시하며, 자동으로 필요한 조치를 취함으로써 그것을 중단시키는 보안 솔루션이다. 이것은 단순히 네트워크 단에서의 탐지가 제공하지 못하는 각종 서버를 위해 알려지지 않은 공격까지도 방어 할 수 있는 실시간 침입방지 시스템으로 OS 레벨에서 실시간 방어와 탐지 기능을 제공한다[1].

이 IPS 이전에도 방화벽이나 IDS 같은 보안 솔루션은 많았지만 많은 문제점이 제기 되었다. 그 문제점들은 [표 1]에 정리 하였다.

[표 1] 기존 시스템의 문제점

기존 시스템	문 제 점
방화벽	▪ 방화벽 자체적으로 제공되는 서비스가 제한적임 ▪ 특정 분야에 대한 보안 기능의 집중화로 인한 역기능 발생
IDS	▪ 몇몇 알려지지 않은 공격에 대한 탐지가 곤란 ▪ 트래픽 분석을 위해 생성된 많은 양의 데이터 처리를 위한 문제 ▪ 침입 탐지의 오판 가능성

이러한 문제점을 해결하기 위해 IPS 는 공격적인 신호를 찾아내 네트워크에 연결된 기기에서 수상한 활동이 이뤄지는지를 감시하고 자동으로 모종의 조치를 취함으로써 그것을 중단시키는 솔루션이다.

이러한 IPS 가 갖추어야 할 요건으로서 첫째, 고도의 정확성을 지녀야 하고 둘째, 단순한 탐지가 아닌 방지를 목적으로 하며 셋째, 광범위한 공격 방지 적용을 해야 하며 넷째, 관련된 모든 트래픽을 분석 해야 한다. 다섯째 고도의 세밀한 탐지 및 대응이 필요하다. IDS 는 잘못된 탐지 경보를 남발해 진짜 공격을 놓칠 위험을 감수하면서 상당한 자원을 오 경보 추적에 할 수도 있기 때문에 제한된 인원과 인프라에 심각한 영향을 주었다. 여섯째, 확장 가능한 위험 관리 능력을 갖추어야 한다. 마지막으로 최대 센서 가동 시간은 신뢰도 높은 센서로 방화벽, 라우터, 교환기 등과 비슷한 시간 동안 가동할 수 있어야 한다.

2.2 .1 TimeSys Linux

TimeSys Linux 는 TimeSys 사에서 만든 Embedded System 을 위한 Linux Kernel 으로써 최근의 주요 Embedded Linux 들의 요소들을 포함하고 있다. TimeSys Linux 의 특징은 다음과 같다.

TimeSys Linux 는 필요에 따라 커스터 마이징 하기 쉽고, 커널 지연이 낮으며, 정확한 클럭과 타이머를 제공하여 실시간 기능을 잘 지원한다. 또한 커널 2.6 기반으로 제작되어 O(1) scheduling 과 periodic scheduling 을 지원하고, 정확한 CPU 자원 분배와 네트워크 대역폭 분배를 지원 및 멀티미디어 지원 등 Standard Kernel 2.6.x 의 장점을 상당부분 지원한다 [2].

2.2.2 uClinux

uClinux 는 MMU 가 없는 프로세서를 위해 수정된 Linux Kernel 로, 2.0.x, 2.4.x 버전까지는 독립적으로 개발 되어왔으나 2.6.x 버전부터는 m68knommu 아키텍처를 시작으로 Standard Linux Kernel 에 포함되고 있다. 주요 특징으로는 MMU(Memory Management unit) 지원 부분이 배제되므로 이에 대한 코드부분이 포함되지 않고, 생성되는 kernel 이미지가 작다. 또한 관련 어플리케이션도 상대적으로 30 ~ 50% 정도 작아지는 특징이 있다. 또한, 일반적인 DRAM 으로 경유 하지 않고 바로 ROM, 또는 Flash 에서 직접 application 의 실행 가능하고, Memory Management 관련 몇 가지 예외를 제외 하고는 기존의 Linux API 대부분을 사용이 가능하므로 application 개발에 있어서도 편리하다.

본 논문에서는 standard kernel 2.4.x 기반의 uClinux 와 standard kernel 2.6.x 기반의 uClinux 를 성능테스트에 포함했다.[3]

2.2.3 Standard Linux Kernel

Embedded Linux kernel 과 기존의 엔터프라이즈급의 Linux kernel 과의 성능 비교를 위해 standard Linux kernel 의 2 개 버전을 포함 했다.

Kernel 2.4.x 의 경우 SMP(Symmetric Multi Processing) 시스템에서의 성능이 향상되었고, 사용자와 그룹의 개수를 32 비트로 늘렸다. 물리적인 메모리 사용을 64GB 까지 늘리고, 2GB 파일 크기제한이 없어졌다. 또한 생성 할 수 있는 프로세스의 수가 무한대로 바뀌는 등 많은 특징을 가지고 있다.

다음으로 최근에 공식적으로 많이 적용이 되고 있는 2.6.x 버전의 경우 기존의 2.4.x 에서 많은 발전이 있었는데, 먼저 MMU 가 없는 프로세서의 지원 등 적용되는 Platform 의 범위가 매우 넓어졌다, 또한 엔터프라이즈급 서버지원을 위한 NUMA(None-Uniform Memory Access) Architecture 를 포함하고 있고, 하이퍼스레딩 기능 지원 또한 포함하고 있다. 그리고 ALSA 와 같은 멀티미디어 지원도 강화 되었고, IPsec(IPsecurity), IPv6 등과 같은 최신 네트워크 프로토콜도 본격적으로 지원하고 있다.

2.2.4 Coyote Linux

Coyote Linux 는 Vorteck 사에서 개발한 운영체재로써 아주 작은 용량의 배포판 리눅스 이다. Coyote Linux 는 주로 local network 에서 여러 대의 컴퓨터를 연결해 사용하면서 인터넷을 공유할 때 라우터나 방화벽의 구축을 위해 특화되어 쓰이는데, 시스템 구축을 위한 하드웨어 요구사항이 상대적으로 낮고, 최신의 주요 인터넷 프로토콜을 지원한다. 또한 QoS 를 지원하며, 최소한의 시스템 구성 시 hard drive 나 CD-ROM 등이 필요치 않다[4].

2.3 BYTE Unix Bench Marks(version 4.1.0)

Uinxbench 라고도 많이 알려져 있는 BYTE Unix bench mark 4.1.0 은 User Level Bench mark 수준의 benchmark 툴로서 Linux 에서 가능한 표준 bench mark 프로그램 중 하나이다. 주요 성능 측정 항목은 파일 카피, 파이프 라인 생산, 프로세스 생성, 쉘 스크립트 수행능력 등, 시스템의 하드웨어, 드라이버, 운영체제, 컴파일러 전반에 걸쳐 성능 테스트를 수행

한다. 본 논문에서는 Unixbench 로 사용한다.

3. 설계

3.1 개발환경

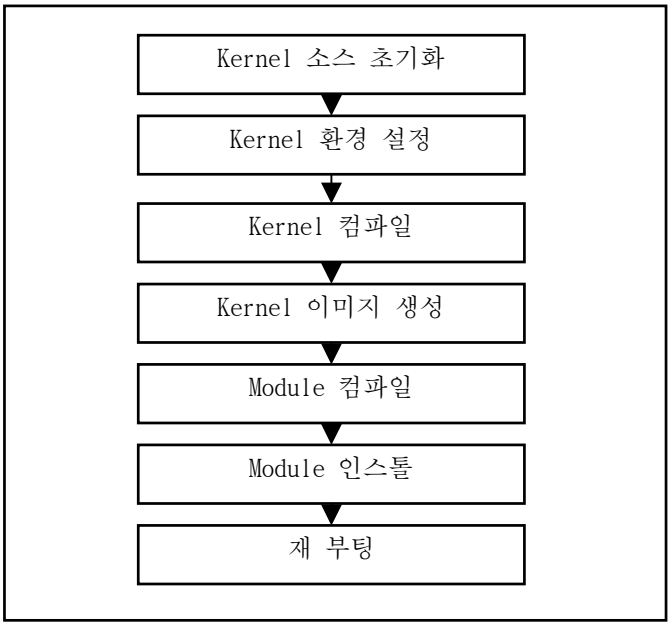
[표 3] 개발 시스템 환경

	사 양
CPU	Intel Pentium III Celeron 700MHz
RAM	512RAM
Complier	GCC 3.2.2
Testing Mode	Single Mode

성능 평가를 위한 테스트 환경을 [표 3]과 같이 구축했다. 평가의 공정성을 위해 Single Mode 에서 성능 test 를 수행 했다.

3.2 Kernel Image 생성

시스템에 성능 테스트를 할 Kernel Image 를 생성하고 적용하는 순서는 [그림 1]과 같다.



[그림 1] Kernel Image 생성 과정

Kernel 환경설정 시 부팅을 위한 최소 옵션만을 선택해서 컴파일을 했다.

Kernel 2.6.x 기반의 Linux 들은 Kernel 컴파일 전에 module init tools 를 설치 해야 한다. Kernel 2.6.x 부터는 modprobe 가 참조하는 파일이 달라지는데, module init tools 이 새로운 파일로 변환해 준다. 본 논문에서는 Timesys Linux kernel 과 uClinux 들 중 2.6.x 기반의 Kernel, 그리고 Standard Linux Kernel 2.6.x 가 포함 된다[5].

3.3 성능 평가

성능테스트 프로그램은 테스트 할 Linux Kernel 을 Single Mode 로 부팅 후 각각 컴파일 하여 프로그램을 실행 시켰다. 하나의 Kernel 성능 테스트를 완료하는데 평균적으로 1 시간 정도 소요되었다. 주요 성능 비교 항목은 [표 4]에 나타내었다.

[표 4]주요 성능 비교 항목 및 BASELINE 값

	BASELINE
File Copy	1655.0
Process Creation	126.0
Shell Scripts	6.0
System Call Overhead	15000.0

UnixBench 에서는 BASELINE 의 수치를 기준으로 하여 Benchmark 결과를 Index 로 계산하여 표시한다.

[표 5] 파일 복사시 성능결과

	Result	Index
uClinux 2.4.x	47449.0	286.7
Coyote Linux	47114.0	284.7
Timesys linux	48783.0	294.8
Standard Kernel 2.4.x	47371.0	286.2
uClinux 2.6.x	46458.0	280.7
Standard Kernel 2.6.x	44183.0	267.0

[표 5]의 File Copy 는 변수 버퍼를 사용하여 하나의 파일에서 다른 파일로 옮길 때의 결과 값을 측정한다. 버퍼 사이즈가 작으면 시스템 콜이 점유하는 시간이 늘어나 오버헤드가 생기게 된다. 한번에 4kb, 1kb, 256byte 씩 3 가지 측정 방법이 있는다. 본 논문에서는 한번에 256byte 를 옮기는 방법으로 성능 측정을 했다.

[표 6] Process Creation

	Result	Index
uClinux 2.4.x	8001.9	635.1
Coyote Linux	8006.2	635.4
Timesys linux	5345.7	424.3
Standard Kernel 2.4.x	4465.2	354.4
uClinux 2.6.x	4672.9	370.9

Standard Kernel 2.6.x	4325.8	343.3

[표 6]의 Process Creation 은 일정 시간 동안 생성되는 Process 의 수를 나타낸다.

[표 7] Shell Scripts

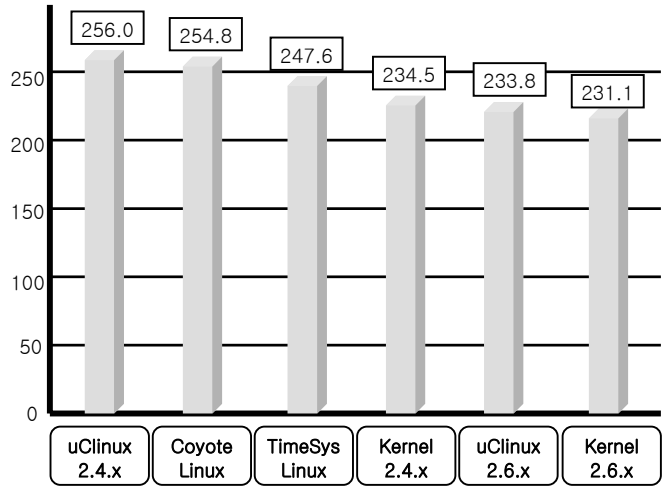
	Result	Index
uClinux 2.4.x	243.0	405.0
Coyote Linux	243.0	405.0
Timesys linux	207.0	345.0
Standard Kernel 2.4.x	205.0	341.7
uClinux 2.6.x	211.0	351.7
Standard Kernel 2.6.x	212.0	353.3

[표 7]의 Shell Scripts 는 여러 개의 Shell Script 들을 동시에 실행 시킨 후 모든 작업이 종료되기까지의 기간을 평가한다. 본 논문에서는 8 개의 Shell Script 를 동시 실행시키고 측정하였다.

[표 8] System Call Overhead

	Result	Index
uClinux 2.4.x	323979.8	216.0
Coyote Linux	324543.2	216.4
Timesys linux	537775.9	358.5
Standard Kernel 2.4.x	317846.6	211.9
uClinux 2.6.x	517839.5	345.2
Standard Kernel 2.6.x	514755.6	343.2

마지막으로 System Call Overhead 는 일정 시간 동안 System Call 호출에 따른 Overhead 크기를 나타낸다.



[그림 2] Final Score

UnixBench 에 의한 Benchmark 결과로써, Final Score 를 위 [그림 2]와 같이 나타내었다. Final Score 를 비교하면 한가지 특이한 점을 볼 수 있는데, 일반적으로 Kernel 2.6.x 기반의 Linux 의 성능이 Kernel 2.4.x 기반의 Linux 들보다 전체적인 성능면에 있어서 좀 더 낮은 수치로 나타났음을 알 수 있다. 이는 Kernel 2.6.x 기반의 Linux 는 Kernel 에서 자체적으로 지원하는 기능들이 커서 이 부분을 지원하기 위해 상대적으로 시스템 자원을 Kernel 2.4.x 기반 보다 더 많이 쓰기 때문이다.

4. 결론

본 논문에서는 동일한 하드웨어 플랫폼에서 Standard Linux Kernel 과 uClinux, Timesys Linux, Coyote Linux 에 대한 Kernel 의 성능을 측정하고 결과에 대해서 비교 분석하였다. 실험 결과 다른 Linux Kernel 보다 uClinux 가 더 좋은 결과를 보였다. 이 같은 결과는 MMU 를 제거함으로써 커널 소스 자체가 경량화 되었고 또한 MMU 가 제거됨에 따라 메모리 연산 시간이 감소되었기 때문이다.

본 논문의 실험 결과를 바탕으로 제시된 시스템에서 실시간 네트워크 감시 및 유/무선 통합 망에 따른 새로운 공격차단 등, 다양한 네트워크 시스템 보안요소를 적용하는 연구를 계속 진행할 것이다.

참고문헌

[1] 권수갑, "*IPS(Intrusion Protection System)동향*", 전자부품연구원 전자정보 센터, 2003
[2] TimeSys FreeBSP, http://www.timesys.com
[3] uClinux Developer Forum, http://www.ucdot.org
[4] Coyote Linux Reference Manual.
[5] Linux Documentation Project, http://mirror.kernel.org/LDP/
[6] Craig Hollabaugh, Ph.D, "Embedded Linux", Addison-Wesley, 2002.
[7] Embedded Linux Consortium, http://www.embedded-linux.org
[8] Karim Yaghmour, "Building Embedded Linux Systems", OReilly & Associates, Inc. 2003