개선된 스냅샷 부트를 이용한 임베디드 리눅스의 빠른 부팅 기법

(A Fast Booting Technique using Improved Snapshot Boot in Embedded Linux)

박 세 진 † 송 재 환 ††
(Sejin Park) (Jaehwan Song)

박 찬 익 †††
(Chanik Park)

요 약 본 논문에서는 임베디드 리눅스를 운영체제로 사용하는 개인용 휴대 임베디드 기기에서 부팅 시간을 단축시키기 위해 기존의 snapshot boot을 개선한 기법을 소개한다. Snapshot boot는 현재 수행중인 컴퓨팅 작업들에 대한 suspend/resume 작업을 기반으로 부팅 시간을 단축하고자 하는 기법인데, resume수행 시 bootloader가 기본적인 device 초기화 작업을 수행하고 직접 snapshot image를 원래 주소로 복사시켜 시스템을 복원함으로써 부팅 시간을 단축시켰다. Snapshot boot 기법의 문제점으로는 resume 동작을 수행할 때 snapshot image를 원래 주소로 복사하는데 많은 시간이 소요된다. Improved snapshot boot 기법은 suspend 작업 수행 시 모든 페이지를 대상으로 snapshot image를 만들지 않고 일부 페이지를 대상으로 snapshot image를 만들고 나머지 페이지들은 별도의 swap area에 따로 저장함으로써 부팅 시 전체 페이지를 복사하지 않고 snapshot image로 만들어져 있는 일부의 페이지만을 복사하게 되어 전체 부팅 시간을 단축한다. 실험을 통해 suspend image가 2982 페이지일 때 약 30%의 부팅시간이 단축됨을 보였다. 이는 swap-out 시킨 페이지의 양에 비례하여 단축된다.

키워드 : 스냅샷, 부팅, 부트, 임베디드, 서스펜드, 복원, 스왑

Abstract In this paper we propose a fast booting technique based on Improved snapshot boot in embedded Linux, widely adopted in personal devices such as PDA and mobile phones. The existing Snapshot boot technique tries to create a snapshot image at the time of suspend, and later load the entire snapshot image into the system memory at the predefined location with the help of a bootloader at the time of resume. Since a bootloader has to copy the entire snapshot image into the predefined memory to resume the previous suspended computing state, a little bit long time is required to resume. Improved snapshot boot does not create a snapshot image consisting of whole memory pages at the time of suspend, thus resulting in smaller snapshot image than the existing snapshot boot. The remaining pages are in the swap area. The resulting smaller sized snapshot image enables much faster booting latency. Through the experiment, we can see the booting latency is reduced almost 30 % with suspend image of 2982 pages. This result depends on the amount of swap-out pages.

Key words : snapshot, booting, boot, embedded, suspend, resume, swap

1. 서 론

최근 들어 MP3 플레이어, 모바일 폰, PMP, PDA등 다양한 임베디드 장비가 개인 휴대용 기기들로 범용화되었고 이제는 생활하는데 중요한 필수품으로 자리매김하게 되었다. 부팅 시간은 이러한 기기들의 제품 경쟁력을 강화하는 중요한 요소로 부각되고 있다. 이러한 부팅 시간을 줄이기 위해 다양한 방법들이 논의되고 있다. 대표적인 몇 가지를 나열해 보면 먼저 Kernel XIP기법이 [1] 소개되었다. Kernel XIP(eXecute In Place)은 현재 uClinux에서 구현되어 있는 기법으로 커널을 실행 가능한 롬 영역인 XIP에 위치시키는 방법인데 XIP의 장점은 동일한 프로그램을 여러 번 실행할 때 텍스트 세그먼트를 복사할 필요가 없다는 것이다. 실제로 텍스트 세그먼트는 플래시 메모리 상에 존재할 수 있기 때문에 시스템의 RAM에 복사될 필요가 없기 때문에 부팅 시에 일어나는 다양한 복사로 인한 오버헤드 시간을 줄이게 된다.

이처럼 직접 커널의 복사시간을 줄여 부팅 속도를 개선하는 방법도 있지만, 커널의 초기화 시간을 많이 소모하는 부분들에 대해 처리를 미루는 방법으로 부팅시간

· 본 연구는 교육인적자원부의 BK21 사업 과 정보통신 연구 진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음 (IITA-2008-C1090-0801-0045)
· 이 논문은 제34회 추계학술대회에서 '개선된 스냅샷 부트를 이용한 임베디드 리눅스의 빠른 부팅 기법'의 제목으로 발표된 논문을 확장한 것임

† 학생회원 : 포항공과대학교 컴퓨터공학과
baksejin@postech.ac.kr
†† 학생회원 : 포항공과대학교 정보통신학과
gsrjj2s@postech.ac.kr
††† 종신회원 : 포항공과대학교 컴퓨터공학과 교수
cipark@postech.ac.kr

논문접수 : 2007년 12월 6일
심사완료 : 2008년 5월 28일

을 줄이는 방법도 있다. Delayed Devices Probing 기법이[2] 예가 된다. 부팅과정에서 많은 시간을 소모하는 IDE Device Probe를 부팅 이후에 시스템이 미루어 실행 함으로 부팅 시간이 그만큼 줄게 된다.

본 논문에서는 Snapshot Boot 기법과[3] 이를 좀 더 개선한 Improved Snapshot Boot기법을 제시하고 있다. Snapshot Boot 기법을 간략히 설명하면 현재 수행중인 프로세스들을 정지 시키고, 프로세스들, 각종 메모리 자료들 및 레지스터 값들을 보조기억장치에 snapshot image 형태로 저장해 두고, 다음 부팅 시 보조기억장치에 저장되어 있는 snapshot image를 불러들여 레지스터 값, 각종 메모리 자료 등을 복원하여, 별 다른 커널의 초기화 작업 없이 빠르게 부팅이 이루어지도록 하는 것이다. 이 기법은 현재 사용중인 모든 메모리 페이지를 snapshot image 형태로 저장하고 복원하는 방식으로 되어 있는데, 프로세스의 수가 많아지면 페이지의 저장 및 복원 시간도 비례하여 증가한다. Improved Snapshot Boot 기법은 snapshot image를 만들 때 전체 페이지를 저장하지 않고 snapshot image의 일부를 별도의 swap area에 두고, 일부의 snapshot image 만으로 부팅함으로써 부팅 시간을 단축시킨다. 부팅 후 swap area에서 나머지 페이지를 가져오게 함으로써 프로세스가 많아져도 부팅시간이 비례해서 크게 늘어나지 않는 장점을 가지게 된다.

2. Snapshot Boot

snapshot boot와 improved snapshot boot는 리눅스 커널이 지원하는 software suspend 기법에 기반을 두고 있다. Snapshot boot와 improved snapshot boot의 이해를 돕기 위해 먼저 software suspend/resume 절차를 살펴보고자 한다.

2.1 Software Suspend/Resume

Software Suspend/Resume 방식은 현재 각 상태들을 저장시킨 후 다음 부팅 시 별도의 로딩 과정 없이 커널 및 User Application이 한 번에 일어나는 방식이다. Suspend가 시작되면 사용자 프로세스와 kernel 태스크들을 현재 상태에서 더 진행되지 않도록 중지시키고 불필요한 유휴 메모리를 해제시킨다.

Snapshot image는 현재 수행중인 각 프로세스의 페이지들로 구성되어 있기 때문에 불필요한 페이지들을 최대한 해제시켜 Snapshot image의 크기를 줄인다. 그 후 각 Devices를 suspend시키고 전원을 내리고 마침내 프로세서의 상태와 각 프로세서 레지스터들을 저장한다. 이 상태가 나중에 Resume 수행 시 복원되는 시점이 된다. 실제 snapshot Image를 저장하기 위한 메모리 영역을 할당 받고 할당 받은 영역으로 메모리 내용(페이지)을 복사한다. 복사작업을 마치면 해당 이미지를 보조기억장치에 기록하기 위해 다시 프로세서와 Device를 복원한다.

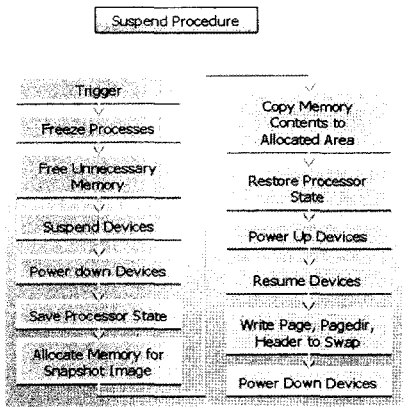

그림 1 Software Suspend Process

활성화된 Device를 통해 저장되어 있던 페이지, 페이지 디렉토리, 헤더이미지를 보조기억장치(swap)로 기록한다. 기록이 끝나면 모든 작업이 완료 되었으므로 각 Device의 전원을 내리고 시스템을 종료하게 된다. 지금까지의 내용을 세부적으로 도식화 하면 그림 1과 같다.

Software Resume의 수행 절차는 다음과 같다.

커널 파라미터를 읽어 snapshot image가 저장되어 있는 디바이스를 인식해 해당 디바이스에서 snapshot image를 읽어온다. 우선 해당 디바이스에 저장되어 있는 이미지가 올바른 이미지인지 snapshot image signature를 체크하고 snapshot image 정보가 들어있는 header를 읽는다. 올바른 이미지로 확인이 되면 해당 snapshot image를 로드 하기 위한 메모리 공간을 할당한다. 이 영역에 snapshot image로부터 페이지 디렉토리와 페이지 내용을 저장한다. 아직은 resume하기 전의 커널이 수행 중이므로 수행중인 프로세스를 정지하고 불필요한 메모리를 해제한 후 각 디바이스들을 suspend시키고 전원을 내린다. 그리고 Snapshot image resume이 실패할 경우를 대비해서 현재 프로세서의 상태 및 레지스터를 저장한다. Resume 작업을 위해 메모리에 복사해 둔 Snapshot image를 원래의 주소로 재 복사하게 됨으로써 메모리의 복원이 끝나고 snapshot image생성시의 프로세서 레지스터와 프로세서 상태를 복원함으로써 프로세서의 복원을 끝내고 마지막으로 각 디바이스들을 resume하고 정지되어있던 프로세스들을 재 기동시킨다. 지금까지의 내용을 도식화하면 그림 2와 같다.

2.2 Snapshot Boot

Snapshot Boot은 Software Suspend / Resume 기법을 보완한 Fast boot 기법이다. 앞서 설명한 Soft-

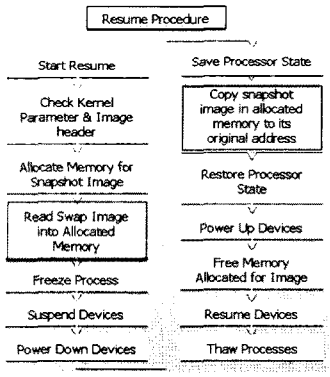

그림 2 Software Resume Process

ware Suspend / Resume 기법은 우선 Resume을 리눅스 커널 자신이 수행하기 때문에 resume을 수행 할 수 있을 만큼 커널의 로드 및 초기화가 이루어 져야 한다. Snapshot image에 이미 커널 image도 저장되어 있고, 커널 제어 흐름조차 복원시키기 때문에 resume을 수행하기 위해 커널을 로드 하는 것 자체가 하나의 큰 오버헤드가 된다. 또한, Software Suspend / Resume 기법은 메모리 복원 시 두 번의 복사과정이 일어나게 되는데 ((1).임의영역 메모리 할당 후 snapshot image 복사, (2).임의영역에 복사된 snapshot image를 원래 주소로 재 복사) 이는 한번의 수행으로 끝낼 수 있지만 일단 resume을 수행시키는 커널을 로드해야 하기 때문에 두 번의 복사를 피할 수 없게 된다.

Snapshot boot는 이러한 단점을 bootloader와의 연계를 통해 해결 하였다. Resume을 위해 커널을 로드 할 필요 없이 bootloader가 기본적인 Device 초기화 작업을 수행하고 직접 snapshot image를 원래 주소로 복사시켜 시스템을 복원하게 된다. 이러한 절차를 통해, resume작업을 위한 커널의 로드가 필요 없어지며, 그래서 메모리 복사가 두 번 일어날 당위성도 없어지게 된다. 아래의 그림 3은 Software suspend/resume 기법과 snapshot boot 기법의 부팅 flow를 비교하여 도식화한 그림이다.

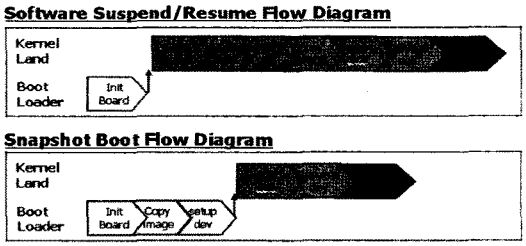

그림 3 Software Suspend/Resume vs Snapshot boot

3. Improved Snapshot Boot

3.1 부팅 소요시간

Snapshot boot의 부팅 소요시간은 다음과 같이 나타낼 수 있다.

$$Startup\ time = \sum I/O\ Initialization + \sum Image\ copy + \sum Kernel\ Resume$$

Snapshot boot는 Kernel의 로드 없이 bootloader에서 드라이버 초기화 및 snapshot image 복사를 수행하게 된다. 이 부분에서 Snapshot boot의 대부분의 시간을 소비하게 된다. 이 두 부분의 수행시간을 측정한 결과는 표 1과 같다.

표 1의 결과를 살펴보면 Image Copy 시간이 전체 Snapshot boot의 대부분을 차지함을 알 수 있다. 또한 이 시간은 이미지로 저장되어 있는 page수에 비례하여 증가한다.

표 1 snapshot boot time consuming

Part	Driver Initialization	Image Copy (2300pages)
Time	1 sec	4 sec

3.2 Improved Snapshot Boot의 특징

Improved Snapshot Boot은 snapshot boot 기법에서 많은 시간을 소비하는 Image Copy 시간을 줄여 전체 부팅시간을 줄이는 기법이다. Snapshot Boot 기법은 수행중인 페이지가 많으면 이미지 전체의 크기도 커지게 되고 이는 Resume 연산 수행 시 Image Copy 시간이 길어지는 결과를 초래한다.

Improved Snapshot Boot는 Suspend 작업 수행 시 모든 페이지를 대상으로 snapshot image를 만들지 않고 일부 페이지를 대상으로 snapshot image를 만들고 나머지 페이지들은 별도의 swap area에 따로 저장한다. 결과적으로 부팅 시 전체 페이지를 복사하지 않고 snapshot image로 만들어 져 있는 일부의 페이지만을 복사 하게 되어 전체 부팅 시간이 줄어들게 된다. 그림 4는 이 과정을 도식화한 것이다.

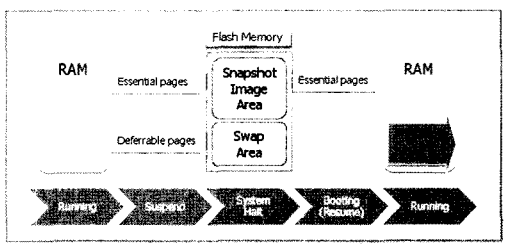

그림 4 Improved Snapshot boot

위 그림 4에 나타난 Improved Snapshot Boot의 Suspend 및 Resume 전체의 수행절차는 다음과 같다.

1. Suspend 수행 시 페이지들을 두 영역에 나누어 저장 (Snapshot Image Area, Swap Area)
2. 부팅시 Snapshot Image Area에 위치한 Image로 resume 수행
3. 부팅이 끝나고 사용자의 요청이 들어오면 Swap Area에서 해당 페이지를 복원

3.3 Improved Snapshot Boot의 구현

Improved snapshot boot는 snapshot boot 기반으로 구현이 된다. Suspend시 User process 페이지를 swap area에 따로 저장을 하는데 이때 리눅스의 swap-out mechanism으로 처리한다. 리눅스의 swapping mechanism을 이용하여 저장을 하면 resume작업 수행 시 장점이 생기는데, 해당 page에 대해 별도의 추가작업을 할 필요 없이 swap area에 저장되어 있던 페이지가 리눅스의 Page Fault Handler에 의해 자동적으로 처리된다[4].

즉 리눅스 커널의 Swap-in 과 Swap-out을 이용해 큰 변화 없이 개선된 사항이 적용될 수 있도록 하였다. 이때 Snapshot boot에서는 mtd영역 하나에 Snapshot image를 저장하고 있는데, 이를 두 부분으로 나누어 구현한다.

4. 실험 결과

4.1 실험 환경

본 논문에서는 임베디드 시스템을 위한 Software suspend를 평가하기 위해 OMAP 5912 Starter Kit이 사용되었다. 이 보드는 ARM926EJ 프로세서를 장착하고 있고 32Mbytes의 SDRAM과 32Mbytes의 NOR Flash의 메모리를 장착하고 있다. 사용된 리눅스는 커널 버전 2.6.11에 해당 보드에 맞게 패치를 적용하여 사용하였고 Snapshot boot Patch를 적용하였다. 각 실험은 top 명령을 background로 수행시키고 suspend를 수행하여 resume 후에 top 명령이 background로 수행중인 지를 확인하고 각 부팅 기법 별 수행 속도를 측정하였다.

4.2 Software Suspend/Resume 결과

Software Suspend/Resume 기법에서 suspend to disk가 Triggering 될 때 메시지가 콘솔로 방출되고 Power off 되고 마침내 시스템이 Halt 된다. 이때 각 메모리상의 페이지들이 하나의 이미지로 만들어져서 다음 부팅 시 메모리로 복구된다. 다음의 그림 5는 그 과정을 보여준다.

Software Suspend/Resume 기법으로 부팅할 때 시스템은 snapshot image를 읽고 resume을 수행한다. 이 과정에서 System image copy가 storage에서 working buffer로 1 회, 또 working buffer에서 final address로 1회 총 2회가 이루어지고 이때 의 복사시간이 오버헤드

```
# mount -t sysfs none /sys
# mkswap /dev/mtdblock3
Setting up swapspace version 1, size = 31191040 bytes
# swapon /dev/mtdblock3
[   39.070765] Adding 30456k swap on /dev/mtdblock3.  Priority:-1 extents:1
# echo disk > /sys/power/state
[   49.843650] Stopping tasks: ====|
[   49.847835] Freeing memory... done (321 pages freed)
[   50.090346] Writing data to swap (1238 pages)... done
[  103.432615] Writing pagedir (5 pages)
[  103.439098] S|
[  106.082171] flush MTD
[  107.344103] Powering off system
[  107.347524] System Halted
```

그림 5 Suspend to disk

```
[    1.951658] NET: Registered protocol family 1
[    2.005821] Relocating pagedir ..::|
[    2.054589] Reading image data (1271 pages): 100% 1271 done.
[    5.253011] Stopping tasks: ===|
[    5.256806] Freeing memory... done (0 pages freed)
[    5.420427] Restarting tasks... done
[    8.886462] swsusp: Mark swsusp again
#
# ps
  PID  Uid      VSZ Stat Command
    1 root     1888 S    init
    2 root          SWN [ksoftirqd/0]
    3 root          SW< [events/0]
    4 root          SW< [khelper]
    9 root          SW< [kthread]
   17 root          SW< [kblockd/0]
   63 root          SW  [pdflush]
   64 root          SW  [pdflush]
   66 root          SW< [aio/0]
   65 root          SW  [kswapd0]
  652 root          SW  [kseriod]
  780 root          SW  [mtdblockd]
  786 root          SW  [pccardd]
  716 root          SW< [rpciod/0]
  722 root     1888 S    -sh
  726 root     1888 T    top
  727 root     1884 R    ps
#
```

그림 6 Software Suspend/Resume boot

로 작용된다. 위의 그림 6은 그 과정을 보여준다. 그림에서 보이는 것처럼 커널이 이미 로드가 되어있고 resume 수행 후 새로운 커널로 제어권이 넘어간다.

4.3 Snapshot Boot 결과

Snapshot Boot 기법은 Bootloader가 커널의 로드 없이 직접 snapshot image를 swap에서 original 메모리 주소로 바로 copy함으로써 부팅시간을 단축시켰다. Snapshot boot가 시작되고 시스템은 이전의 suspended 시스템 상태로부터 resume된다. 아래 그림 7은 그 과정을 보여준다.

```
bootss: OMAP MPU timers initialized
        pages to copy: 1271 (0x000004f7)
bootss: copy image done.
bootss: jumping to kernel resume point: 0xc015ee4c
|    3.280391] Restarting tasks... done
|    4.595759] swsusp: Mark swsusp again
# ps
  PID  Uid      VSZ Stat Command
    1 root     1888 S    init
    2 root          SWN [ksoftirqd/0]
    3 root          SW< [events/0]
    4 root          SW< [khelper]
    9 root          SW< [kthread]
   17 root          SW< [kblockd/0]
   63 root          SW  [pdflush]
   64 root          SW  [pdflush]
   66 root          SW< [aio/0]
   65 root          SW  [kswapd0]
  652 root          SW  [kseriod]
  780 root          SW  [mtdblockd]
  786 root          SW  [pccardd]
  716 root          SW< [rpciod/0]
  722 root     1888 S    -sh
  726 root     1888 T    top
  727 root     1884 R    ps
#
```

그림 7 Snapshot Boot

4.4 Improved Snapshot Boot 결과

Improved Snapshot Boot 기법의 실험은 기존의

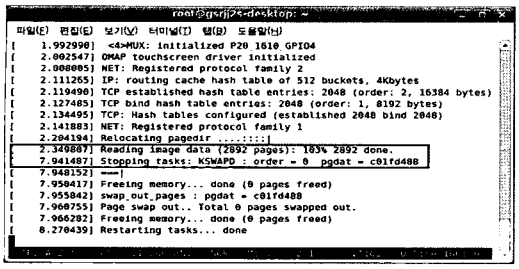

그림 8 Software Suspend & Resume

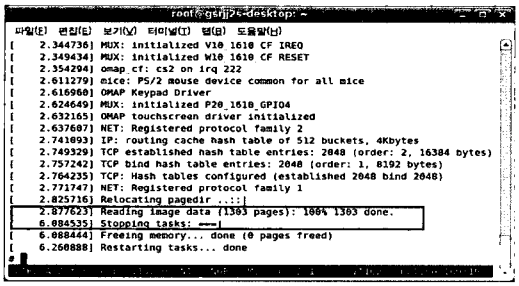

그림 9 Improved Snapshot Boot

Snapshot Boot와의 차이점을 부각시키고 정확한 시간 측정을 위해 Software Suspend & Resume 방식과 비교를 하였는데, Snapshot boot와 Improved Snapshot boot의 차이점은 일부의 페이지만으로 Image만을 만들고 나머지 페이지는 Swap out 시켜 부팅 후에 page fault handler에 의해 Swap in 시켜 부팅 시 일어나는 Image의 크기가 틀려진다는 점이다.

그림 8은 Software Suspend & Resume의 Suspend 과정을 나타내고 있고, 그림 9는 Improved Snapshot boot의 Suspend 과정을 나타내고 있다. 두 방식 모두 같은 프로세스가 실행중인 상황에서 Suspend를 하였으며, Improved Snapshot boot 방식은 일부 page들을 Swap 영역에 따로 저장하여 페이지의 수가 많이 줄어들었다.

위 두 방식에서 시간상의 차이점을 확인하면, Software Suspend & Resume 방식에서 총 2892 페이지를 복사하고 이때 소요 시간은 약 5.6초가 걸리는데 반해, Improved Snapshot Boot에서는 총 1383 페이지를 복사하고 이 때 소요 시간은 약 3.21초로 페이지 수에 비례하게 시간이 걸림을 알 수 있다. 현재 Swap 영역으로 보내는 페이지는 리눅스 커널 2.6의 페이지 회수 정책을 기준으로 정해지며, 페이지를 회수시키는 임계값인 page low의 값을 조절함으로써 이루어진다.

4.5 항목 평가

실험 결과를 살펴보면 두 방식에서 시간차이가 약 4초 정도 나는 것을 확인 할 수 있다. 표 2는 각 방법 별 부팅 시간을 나타낸 것이다.

표 2 Booting Time

Method	Software Suspend&Resume (2892 pages)	Snapshot boot (2892 pages)	Improved Snapshot boot (1383 pages)
Time	8.806 sec.	4.596 sec.	3.192 sec.

Snapshot Boot는 이미 startup된 application의 snapshot image를 저장했다가 boot 시에 restore 함으로써 부팅 시 가장 많은 시간이 소요되는 application의 start-up 시간을 줄임으로써 전체적인 부팅시간을 개선하였다.

Improved Snapshot boot는 Snapshot boot에서 일부분의 페이지만으로 Resume를 수행하기 때문에 만약 실행중인 프로세스의 수가 많아 저장될 page의 개수가 많아 질 경우 Snapshot boot는 page수에 비례하여 시간이 늘어나지만, Improved Snapshot boot는 많은 수의 page들이 Swap 영역으로 들어가기 때문에 두 기법의 부팅 속도차이가 크게 차이가 나게 된다.

5. 향후 연구 계획

본 논문에서 제시한 Improved Snapshot Boot 기법은 suspend 시 일부분의 페이지만으로 snapshot image를 만들어 resume시 일어나는 image 복사시간을 단축시키지만, 부팅 후 복원되지 않은 페이지를 접근하는 프로세스는 Page fault가 일어나고 Page fault handler에 의해 swap area에서 해당 페이지를 가져오는 작업을 해야 하기 때문에 수행속도의 저하가 온다. 이를 최소화하기 위해서는 suspend시 swap-out 시킬 페이지들을 선별하는데 다양한 기준을 적용해서 좀 더 나은 방법을 찾아 봐야 할 것이다.

6. 결 론

Snapshot boot는 기존의 Software suspend/resume 방식을 이용하여 부팅시간을 개선하였으나 이 과정에서 snapshot image를 copy하는데 많은 시간을 소비하게 된다. Improved snapshot boot는 snapshot image 작성시 모든 페이지에 대해 만들지 않고 일부의 페이지만으로 snapshot image를 구성함으로써 결과적으로 resume시의 부팅시간을 단축시킬 수 있게 되었다.

참 고 문 헌

[1] CE Linux Forum (CELF) Kernel XIP http://tree.celinuxforum.org/CelfPubWiki/KernelXIP
[2] 박우람, 나윤주, 박찬익, "지연된 장치탐색을 이용한 부팅시간 향상 기법", 정보과학회 2006 추계학술대회.
[3] Hiroki Kaminaga, "Imporving Linux Startup Time Using Software Resume," 2006 Linux Symposium.
[4] Bovet and Cesati, "Understanding Linux Kernel 3rd Edition," O'REILLY.