Debian GNU/Linux 3.1 ('sarge'), ARM 릴리즈 노트

Josip Rodin, Bob Hilliard, Adam Di Carlo, Anne Bezemer, Rob Bradford (현재), Frans Pop(현재)
<debian-doc@lists.debian.org>

$Id: release-notes.ko.sgml,v 1.7 2005/06/10 12:35:31 jseidel Exp $

Contents

Chapter 1

릴리즈 노트에 새로 추가된 내용

[이 문서의 가장 최신 내용은 항상 http://www.debian.org/releases/stable/releasenotes 에서 구할 수 있다. 여러분이 보고 있는 문서가 한달이상 된거라면 가장 최신 문서로 받아서 보는게 좋다.]

지난번 릴리즈보다 오래된 상태에서 업그레이드하는 정보를 포함한 지난번 릴리즈 노트를 참고하라. 그러한 내용은 문서를 자주 복잡하게 만들어서 단지 Debian GNU/Linux 3.0 ('woody')에서 업그레이드만 이야기 하기로 결정했다. 오래된 릴리즈에서 업그레이드가 필요하다면 지난번 릴리즈 노트를 읽어보라.

Chapter 2

Debian GNU/Linux 3.1에 무엇이 새로운가?

지원하는 아키텍처 목록은 지난번 릴리즈 Debian GNU/Linux 3.0 (’woody’) 뒤로 바뀌지 않았다. 이번 릴리즈에 지원하는 아키텍처 목록입니다.

- Intel x86 (’i386’)
- Motorola 680x0 (’m68k’)
- Alpha (’alpha’)
- SPARC (’sparc’)
- PowerPC (’powerpc’)
- ARM (’arm’)
- MIPS (’mips’ (Big endian)과 ’mipsel’ (Little endian))
- Intel Itanium (’ia64’)
- HP PA-RISC (’hppa’)
- S/390 (’s390’)

포트 상황과 포트와 관련한 특정 내용을 읽어보면 여러분이 쓰고 있는 아키텍처에 대한 정보를 Debian port web pages (http://www.debian.org/ports/arm/)에서 얻을 수 있다.

Debian GNU/Linux 3.1 for the ARM 아키텍처용 Debian GNU/Linux 3.1는 2.4.27 커널버전과 함께 들어간다.

2.1 새 안정판에 들어간 새 내용은?

이 새로운 데비안 릴리즈는 이전 woody보다 더 많은 소프트웨어를 가지고 있다; 9000개가 넘는 새 패키지를 포함하고 있고 대부분 소프트웨어가 업데이트가 되었다: 대부분 6500 개의 패키지가(지난번 릴리즈의 73%)있고 많은 수의 패키지가 여러가지 이유로 없어졌다. 이 패키지들은 업데이트가 안될 것이다. 이 패키지들은 패키지 관리 시스템 프론트 엔드에서 'obsolete'으로 나타날 것이다.

이번 Debian GNU/Linux 데비안 릴리즈에는 많이 발전한 XFree86 4.3 릴리즈가 들어있는데 더 많은 종류의 하드워어를 지원하고 더 나은 자동검색 지원과 Xinerama와 3차원 가속기능과 같은 더 나은 기능을 지원하고 있다.

Debian GNU/Linux은 어떤 새로운 릴리즈보다 더욱더 데스크탑 환경에 맞게 맞춰져 있는데 여기에는 GNOME 2.8과 KDE 3.3이 들어있다. OpenOffice.org 1.1의 오피스군이 포함되어 있다. 여기에는 Evolution 그룹웨어와 게임(GAIM) 메신저 클라이언트도 들어가 있다.

새루운 패키지 관리 도구로 콘솔에서 쓸 수 있는 도구로 `aptitude` 쓸 수 있다. `apt-get`보다 더 나은 의존성 문제를 해결해 주고 있다. `aptitude`는 대부분의 `apt-get` 명령행 기능을 지원 하고 있다. 아직도 `dselect`를 쓰고 있다면 패키지 관리도구로 과감히 `aptitude` 써라.

공식 Debian GNU/Linux 배포본은 열세개의 바이너리 CD와 비슷한 양의 소스 CD로 구성되어 있고 DVD 버전도 얻을 수 있다.

2.1.1 새로운 서비스 debian-volatile

새로운 서비스인 *debian-volatile*은 사용자들이 쉽게 오래된 정보를 포함한 안정판 패키지를 업데이트를 빠르고 쉽게 도와준다. 예제는 바이러스 검출기 서명 리스트나 스팸 필터 패턴 모음이 있다. 관리자는 "volatile.debian.net" 아카이브를 써서 수게 "security.debian.org" 아카이브에 접근하고 쉽게 최신의 정보를 헷갈리지 않고 전체 시스템이나 시스템 일부를 안전하게 관리할 수 있다. 자세한 정보는 web page (`http://volatile.debian.net/`)를 참조하라.

debian-volatile은 공식 데비안 서비스가 *아니다*. 알아서 사용하라.

2.1.2 non-US 오래된 것들

sarge 릴리즈에 non-US에 있던 패키지들이 메인 아카이브 로 들어왔다. 만일 "non-us"가 있다면 이를 `/etc/apt/sources.list` 서 없애라.

2.2 설치 시스템에 바뀐점은?

예전 Debian GNU/Linux 설치 시스템은 완전히 새로운 설치 시스템인 `debian-installer`로 대체됐다. 새로운 설치 시스템은 디자인에서 모듈화 되어 있어서 확장이 가능하게 만들어졌다.

설치 시스템에 새로 들어간 몇몇 특징들은 처음 부트 상태에서 패키지를 설치하기 위해 `aptitude`를 선택하면서 USB 플래쉬 장치로 시작할 수 있게 해주는 지원을 담고 있고 `XFS` 파일 시스템과 `LVM` (volume 관리도구)에 대한 지원을 담고 있다.

새로운 데비안 설치 시스템의 자세한 내용을 알려면 사용자들이 데비안 설치 지침서를 첫번째 CD나 http://www.debian.org/releases/stable/installmanual 에서 참조해서 보면 된다.

Chapter 3

새로운 설치

예전 Debian GNU/Linux 설치 시스템인 `boot-floppies`는 새로운 구성성분중심이고 더욱 강력한 `debian-installer`로 대체되었다.

인스톨러는 다양한 설치 방법을 준다. 어떤 설치방법인지는 여러분이 쓰는 아키텍처에 달렸다.

데비안을 새로 설치하려 한다면, 설치지침을 읽어야하는데 이것도 공식 CD안에 들어있다:

`/doc/install/manual/`*`language`*`/index.html`

그리고 인터넷: `http://www.debian.org/releases/stable/installmanual`에서도 볼 수 있다. 데비안-인스톨러에 대한 errata (`http://www.debian.org/releases/stable/debian-installer/index#errata`) 를 찾아볼 수도 있다.

3.1 인기 콘테스트

기술적인 이유로 popcon 패키지는 더이상 기본으로 new sarge 설치에서는 하지 않는다. 다음번 릴리즈에는 수정되서 들어갈 것이다.

popcon은 데비안 프로젝트에 다양한 정보를 준다. 어떤 패키지를 지금 쓰고 있는지와 같은 정보를 주기 때문에 유용할 수 있다. 이 정보는 어떤 패키지를 CD-ROM에 넣을지 결정할 수 있지만 이 정보는 가끔 더이상 관리 안하는 패키지를 입양할지 말지 결정하는데 쓰기도 한다.

popcon는 익명으로 진행된다. 이 패키지를 설치하고 참여하는 사람들에게 감사드린다;이는 데비안을 발전시키는데 도움을 줄 것이다.

Chapter 4

이전 릴리즈에서 업그레이드

4.1 업그레이에 필요한 준비

업그레이드를 하기 전헤 모든 내용을 백업하두고 잃어버릴 수 있는 최소의 데이터나 설정을 백업해둔다. 업그레이드 도구나 프로세스는 매우 믿을만 하지만, 업그레이드 도중 하드웨어가 문제를 발생시켜 시스템에 심각한 문제를 낳을 수 있다.

여러분인 백업할 주요 파일들은 `/etc`, `/var/lib/dpkg` 이고 `dpkg --get-selections \*`의 결과내용이다.

업그레이드 과정 자체는 `/home` 내용을 바꾸지 않는다. 그런데 KDE나 모질라와 같은 어플리케이션들은 기존 사용자 설정을 새로운 디폴트 값으로 바꿔버리는데 이는 사용자가 그 어플리케이션을 새롭게 시작하면 적용된다. 우선 미리 숨김 파일과 디렉토리(".으로 시작하는 파일들")을 사용자 홈 디렉토리에서 백업을 철처히 하라. 사용자들에게 이 내용을 알려주는 일이 중요하다.

여러분이 계확하는 내용의 업그레이드를 모든 사용자에게 알려주고 SSH 사용자들은 거의 그 변화를 알지 못하겠지만 계속 작업하길 바라기 때문에 내용을 알려주면 된다. 다른 주의 사항과 사용자 파티션(`/home`)백업 마운트 해제가 빌요하다면 하라. 재부팅은 필요없을 것이다.

텍스트모드 가상 콘솔에서 로컬에서 작업해야하며(직접 연결된 시리얼 터미널에서), 또는 `ssh`를 통해서 원격에서 작업 할 수 있다.

중요: `telnet`, `rlogin`, `rsh`, `xdm`, `gdm`, `kdm`과 같은 것을 통해서 업그레이드 해선 *안된다*. 이 이유는 업그레이동안 서비스가 끊기고 시스템이 완전히 업그레이드 안된 상세에서 *접근할수 없는* 상황을 만들 수 있다.

업그레이드하는 가장 좋은 방법은 `aptitude`를 쓰는 것이다. 내장 의존성 분석을 통해 부드럽게 업그레이드하고 쉽게 설치해준다.

어떤 패키지를 설치해도 항상 루트로 작업을 해야하는데 `su` 나 `sudo`를 써서 필요한 접근 권한을 얻으면 된다.

4.2 시스템 상태 확인

업그레이드 과정은 "순수하게" woody에서 업그레이드 하는 부분만 다룬다. 여러분 시스템이

woody의 가장 최신 상태에 있다고 가정한다. 확실하지 않으면 'woody 시스템 업그레이드하기' on page 23를 따라하라.

여기서는 `aptitude`가 woody의 버전으로 설치되었다고 가정한다. 이를 확인하려면

```
$ dpkg -l aptitude
```

만일 결과가 "i"로 시작하지 *않으면*, 'aptitude의 woody 버전 설치' on page 23 것처럼 업그레이드를 시작하라.

4.2.1 APT 핀 기능 죽이기

테스팅 이상의 배포본에서 어떤 패키지를 설치하기 위해 APT를 설정 했다면 `/etc/apt/preferences`에 들어있는 APT 핀 설정을 바꿔서 패키지 업그레이드가 새로운 안정 버전으로 되게 해주면 된다. APT 핀기능에 대한 나용은 `apt_preferences(5)` 에서 찾을 수 있다.

4.2.2 패키지 상태 확인하기

업그레이드에 필요한 방법을 뭘 쓰던, 우선 모든 패키지를 검토하는 게 가장 먼저이고 모든 패키지가 업그레이드 상태인지 확인해야한다. 다음 명령은 반정도만 설치됐는지 설정 실패인지 아니면 다른 에러를 가진 패키지인지 확인해준다.

```
# dpkg --audit
```

여러분 시스템에 있는 모든 패키지 상태를 `dselect`, `aptitude`,나 다음과 같은 방법으로 확인할 수 있다

```
# dpkg -l | pager
```

나

```
# dpkg --get-selections > ~/curr-pkgs.txt
```

업그레이드 하면서 모든 홀드 상태는 다 없애길 바란다. 업그레이드에 필요한 패키지가 홀드상태라면, 업그레이드는 실패할 것이다. 홀드 상태인지 아닌지는 다음으로 확인가능

```
# dpkg --get-selections | grep hold
```

패키지를 로컬에서 변형했거나 다시 컴파일한 경우는, 이름을 바꾸지 않고 그 상태로 두고 싶다면 업그레이드 하지 않도록 홀드상태로 두라. 'hold' 패키지 상태는 `aptitude`를 통해서 다음처럼 하면 된다 'hold' 패키지 상태는 `aptitude`를 써서 바꿀 수 있다:

```
aptitude hold | unhold <package>
```

고쳐야할 것이 있다면 '소스 리스트 확인' on page 23에서 말한 것처럼 `sources.list`에서 woody 를 여전히 참고하게 하라.

4.2.3 비공식 소스와 백포트

여러분 시스템에 데비안 패키지가 아닌 패키지가 있다면 업그레 이드 도중에 의존성 충돌로 삭제될 수 있다. 이 패키지들은 `/etc/apt/sources.list`에 추가 패키지로 등록해서 설치했다면 sarge에 맞게 다시 패키지가 나왔는지 확인 하고 데비안 패키지용 소스 라인을 동시에 바꿔서 관리한다.

몇몇 사용자들은 백포트된 비공식인 "좀더 새로운" 이미 woody 시스템에 설치된 데비안에 있는 패키지를 쓸 수 있다. 이 패키지들은 업그레이드 도중 문제를 발생시키기 일쑤있다. 특히 파일 충돌 [1]. '업그레이드 도중 가능한 일들' on page 15섹션에 이런 파일 충돌 문제를 다루는 방법이 나와있다.

4.3 APT 소스 준비하기

업그레이드를 시작하기 전에 `apt` 설정 파일을 `/etc/apt/sources.list`에 맞게 해줘라.

`apt`는 "`deb`" 줄 내용을 따라서 패키지를 검색하고 높은 버전을 먼저 설치하면서 처음줄에 있는 내용에 우선순위를 준다(이런 식으로, 여러 미러가 있을 때, 로컬 하드 디스크, CD-ROM, HTTP/FTP 미러순으로 하게된다).

릴리즈 이름이 코드이름(예를들어, woody, sarge) 와 상태 이름(oldstable, stable, testing, unstable)을 모두 참조 할 수 있다. 코드이름으로 참조하면 새로운 릴리즈가 나왔을 때 절대 놀랄 일이 없다. 또한 여러분이 릴리즈 공고를 계속해서 살펴볼 수 있다는 뜻이기도 하다. 대신 상태 이름을 쓴다면, 릴리즈가 됐을 때 가능한 패키지 업데이트 양을 볼 수 있을 것이다.

4.3.1 APT 인터넷 소스 추가하기

기본 설정은 주 데비안 인터넷 서버에서 받아서 설치하게 돼있고 `/etc/apt/sources.list`를 바꿔서 다른 미러를 쓸 수 있고 여러분 위치에서 가까운 미러를 쓰면 된다.

데비안 HTTP나 FTP 미러 주소는 `http://www.debian.org/distrib/ftplist`에서 볼 수 있다("모든 미러 목록"을 보라). HTTP 미러는 FTP보다 일반적으로 빠르다.

예를들어, 여러분의 가장 가까운 데비안 미러가 `http://mirrors.kernel.org/debian/`라고 가정하자. 웹 서버나 FTP 프로그 램을 써서 이 미러를 찾아보면 주 디렉토리가 다음과 같이 돼있을 것이다:

```
http://mirrors.kernel.org/debian/dists/stable/main/binary-arm/...
http://mirrors.kernel.org/debian/dists/stable/contrib/binary-arm/...
```

`apt`로 이 미러를 쓰려면, 다음 내용을 `sources.list` 에 더하라:

```
deb http://mirrors.kernel.org/debian sarge main contrib
```

[1] 데비안 패키지 관리 시스템 은 정상적으로 패키지를 제거하거나 다른 패키지가 소유한 파일을 대체하지 않는다; 특별히 대체하라고 하지 않으면 그렇게 하지 않는다.

'dists'는 이미 있는 것이고 릴리즈 이름 뒤에 오는 내용들은 다중 디렉토리로 경로를 확장하는데 쓴다.

새로운 소스를 더한 뒤에, sources.list에 있는 이미 있는 "deb"로 시작하는 라인에 #를 맨 앞에 넣어서 사용하지 못하도록 한다.

설치할 모든 패키지는 네트워크로 받아서 /var/cache/apt/archives에 둔다. 다운로드 중에는 partial/이라는 내부 디렉토리에 둔다. 그렇게 해서 설치에 필요한 공간이 우선 충분한지 확인해야한다. 확장된 데비안 설치에는 적어도 300MB 다운로드 자료 공간이 든다.

4.3.2 로컬 미러에서 APT 소스 추가하기

HTTP나 FTP 패키지 미러를 쓰는 대신 /etc/apt/sources.list 을 수정해서 NFS로 마운트된 로컬 디스크를 미러로 쓸 수 있다.

예를 들어, 여러분 패키지 미러를 /var/ftp/debian/ 에 둘 수 있고 다음과 같이 주 디렉토리로 할 수 있다:

```
/var/ftp/debian/dists/stable/main/binary-arm/...
/var/ftp/debian/dists/stable/contrib/binary-arm/...
```

apt로 이를 쓰기 위해선, 다음 내용을 sources.list 에 다음을 더하라:

```
deb file:/var/ftp/debian stable main contrib
```

'dists'는 당연히 들어가는 것이고 릴리즈 이름 뒤에 들어가는 내용은 다중 디렉토리로 가는 경로를 확장하는데 쓴다.

새로운 소스를 더하려한다면, 이미 나와있던 "deb"을 sources.list에서 #을 맨 앞에 더해서 못쓰게 만들면 된다.

4.3.3 CD-ROM에서 APT 소스추가하기

단지 CD를 사용하고자 한다면, /etc/apt/sources.list 에 "deb"앞에 # 처리를 해서 못쓰게 만든다.

/etc/fstab안에 CD-ROM이 /cdrom로 마운트 되는지 확인하라. /cdrom 마운트 지점은 apt-cdrom에게 필요하다. 예를 들면, /dev/hdc 가 CD-ROM이라면 /etc/fstab에 다음 내용을 담고 있어야 한다:

```
/dev/hdc /cdrom auto defaults,noauto,ro 0 0
```

네번째 부분에 defaults,noauto,ro에 사이에 *어떤 공간도 없다.*

확인하려면, CD를 넣고 시도해보라

```
# mount /cdrom   # CD를 마운트 지점으로 마운트 하고
# ls -alF /cdrom # CD의 루트디렉토리를 보여주고
# umount /cdrom  # CD를 언마운트한다.
```

다시 실행한다:

```
# apt-cdrom add
```

각각의 데비안 바이너리 CD-ROM을 가지고 있으면 각 CD를 APT database에 집어넣도록 한다.

4.4 패키지 업그레이드

Debian GNU/Linux 릴리즈 사이에 업그레이드하는 추천하는 방법은 패키지 관리도구인 `aptitude`를 쓰는 일이다. 이 도구는 `apt-get`을 바로쓰면서 가질 수 있는 위험성을 줄이는 도구이다.

루트로 작업을 해서 모든 필요한 파티션 마운트작업을 잊지 말고 다음 명령으로 파티션을 읽기-쓰기 가능하게 만들어라:

```
# mount -o remount,rw /mountpoint
```

`/etc/apt/sources.list`에 APT 소스 항에 'stable' 배포본을 더하고 코드 이름으로 참조하게 하지마라(예를 들어 woody).

`/usr/bin/script` 프로그램을 써서 upgrade 세션 내용을 저장하고 만주게 발생하면 버그 리포트에 정확한 정보를 이 저장 내용을 바탕으로 주라. 기록을 시작하려면, 다음 내용을 쳐넣어라:

```
# script -a ~/upgrade-to-sarge.typescript
```

typescript파일을 `/tmp`나 `/var/tmp`에 두지 마라 이 곳은 업그레이드를 하거나 재부팅을 하면 지워질 수 있기 때문이다.

typescript는 스크롤을 넘어간 화면을 모두 보여주는 기능을 가지고 있다. `alt-F2`를 이용해서 VT2로 넘어간 뒤에 다시 로그인을 해서 `less ~root/upgrade-to-sarge.typescript` 을 해서 파일 내용을 살펴보라.

모든 업그레이드가 끝나면 `typescript`를 `exit`를 입력 해서 `typescript`를 끝내라.

4.4.1 패키지 리스트 업데이트하기

새로운 릴리즈에 필요한 패키지 리스트를 가져와야한다. 이는 [2] 로 할 수 있다:

```
# apt-get update
```

[2] 우리는 `apt-get`을 사용해야하는데 그 이유가 woody 버전의 `aptitude`가 `sources.list`에 새 소스를 더한 뒤에 새 목록 가져오기가 실패할 수 있기 때문이다.

4.4.2 aptitude 업그레이드

업그레이드 테스트는 sarge 버전의 `aptitude`가 업그레이드를 하는 도중에 문제를 `apt-get`나 woody의 `aptitude`를 쓰는 것보다 해결 능력이 좋다. 우선 다음을 사용해서 업그레이드 해야 한다:

```
# aptitude install aptitude
```

변화목록과 이를 확인할 부분을 보게 될 것이다. 제안된 변화 를 유심히 살펴보고 없어지는 패키지를 더 잘보라.

몇몇 경우에 많은 패키지가 없어질 목록에 있다면, 이 목록을 `aptitude`를 따라서 선택된 다른 패키지를 "pre-upgrading" 을 통해 목록을 줄일 수 있다. 예제는 이를 확실히 보여준다. KDE가 깔린 시스템을 써서 업그레이드 테스트 하는 도중, 이 과정이 수많은 KDE 패키지와 perl을 지우는 경우가 있었다. 해결책은 `install aptitude perl`을 `install aptitude`대신 써서 하는 것이다.

4.4.3 Upgrading doc-base

`doc-base`를 설치했다면, 반드시 다른 부분보다 먼저 업그레이드 해야한다. 이유는 동시에 `perl`이 업그레이드 되면 업그레이드가 안된다. 다음 명령을 통해서 설치 되었는지 확인가능하다:

```
# dpkg -l doc-base
```

결과 내용에 "i"라고 설치 정보가 나오면 설치된 것이고 계속 작업 하기 위해서 업그레이드가 반드시 필요하다.

```
# aptitude install doc-base
```

4.4.4 시스템 나머지 업그레이드

업그레이드의 주요 부분을 작업하려면 다음을 실행하라:

```
# aptitude -f --with-recommends dist-upgrade
```

이는 시스템의 완전한 업그레이드를 해줄것인데 이 뜻은 모든 패키지의 가장 최신 버전을 쓰게 해준다는 뜻이고 다른 릴리즈 사이의 패키지 사이에 발생할 수 있는 가능한 의존성 문제를 모두 해결해준다. 필요하다면, 몇몇 새로운 패키지(대개는 새로운 라이브러리나 다른 이름으로 바뀐 패키지)를 설치할 것이고 충돌문제를 일으키는 오래된 패키지를 모두 지워줄 것이다(예를 들어 `console-tools-libs`).

CD-ROM에서 업그레이드를 할 경우, 업그레이드 도중 특정 CD를 몇몇 특정시점에 넣어야할 것이다. 동일한 CD를 여러번 넣을 때도 있다; 이는 CD에 퍼져있는 서로 연관되 패키지 의존성 때문이다.

다른 패키지의 설치 상태를 바꾸는 일이 없으면 업그레이드가 안되는 새로운 버전 패키지는 현재 상태로 "홀드"되어 있을 것입니다. 이는 `aptitude`를 통해서 이 패키지를 선택하거나 `aptitude -f install <package>`를 통해 할 수 있다.

`--fix-broken` (또는 `-f`) 옵션은 `apt`가 깨진 의존성 을 수정하려고 할 때 쓴다. `apt`는 기본적으로 깨진 패키지는 시스템에 그냥 두지 않게 되어있다.

4.4.5 업그레이드 도중 가능한 일들

`aptitude`, `apt-get`, `dpkg`가 작동하는 도중에 다음 에러를 주면서 업그레이드 실패할 수 있다.

```
E: Dynamic MMap ran out of room
```

기본 캐시 공간은 부족하다. 이를 해결할 방법은 `/etc/apt/sources.list`에서 필요없는 부분을 주석처리 하거나 없애고 캐시 크기를 늘리면 해결할 수 있다. 캐시 크기는 `/etc/apt/apt.conf` 에 있는 `APT::Cache-Limit`을 수정하면 늘릴 수 있다. 다음 명령은 업그레이드에 충분하게 만들어줄 것이다:

```
# echo 'APT::Cache-Limit "12500000";' >> /etc/apt/apt.conf
```

여기선 그 파일 안에 변수 모음이 없다고 가정한다.

때로 APT::Force-LoopBreak 옵션을 써서 임시적으로 필수 패키지를 Conflicts/Pre-Depends 순환 때문에 지우게 해줄 수 있다. `aptitude` 는 이를 경고해주고 업그레이드를 말릴 것이다. `-o APT::Force-LoopBreak=1`을 `aptitude` 명령행에 정해서 쓸 수 있다.

시스템 의존성 구조는 수동으로 할 필요가 있을 때가 있다. 이런 경우는 `aptitude`를 쓰거나

```
# dpkg --remove packagename
```

를 하면서 다른 패키지에 안좋은 영향을 주는 패키지를 지우거나

```
# aptitude --fix-broken install
# dpkg --configure --pending
```

극단의 경우 강제 옵션으로 설치를 해야한다.

```
# dpkg --install /path/to/packagename.deb
```

"순수한" woody 시스템에서는 파일 충돌이 일어 나서는 안된다. 하지만 비공식 백포트를 쓰고 있다면 일어날 수 있다. 파일 충돌은 다음과 같이 일어날 수 있다:

```
Unpacking replacement <package-foo> ...
dpkg: error processing <package-name-for-foo> (--unpack):
 trying to overwrite '<some-file-name>',
 which is also in package <package-bar>
```

이런 경우 에러 메시지가 있는 *마지막* 라인에 패키지를 강제로 지워서 해결할 수 있다:

```
# dpkg -r --force-depends packagename
```

모든 문제를 해결하고 나서, 다시 업그레이드를 `aptitude` 명령을 통해서 할 수 있어야한다.

업그레이드 도중, 몇몇 패키지에 대한 설정 문제와 만날 것이다. `/etc/init.d`나 `/etc/terminfo` 디렉토리나 `/etc/manpath.config`에 대한 파일 설정 문제와 만난다면, 파일은 패키지 관리자 버전으로 바뀌고 시스템 일관성 에 맞추기 위해서 'yes'라고 답을 하는게 필요하다. 오래된 버전으로 `.dpkg-old`을 통해서 갈 수 있다.

무엇을 해야할지 모른다면, 패키지나 파일 이름을 적어두고 나중에 정리하라. typescript 파일을 살펴보고 업그레이드 도중 일어나는 일에 대한 정보를 알 수 있다.

4.5 재부팅 하기 전에 해야할 일

`aptitude dist-upgrade`가 끝나면, "정상" 업그레이드가 끝나지만 재부팅을 하기 *전에* 확인 해야할 부분이 있다.

우선 `/usr/share/doc/xfree86-common/README.Debian-upgrade.gz` 파일을 엑스 윈도우 시스템 패키지에 대한 더 자세한 정보니 꼭 읽기 바라고 이는 모든 지난 데비안 릴리즈 사용자에게 해당하는 것이니 꼭 읽기 바란다.

4.5.1 커널 업그레이드

리눅스 커널은 이 과정에서 업그레이드 되지 *않는다*. 이를 하고자 한다면, `kernel-image-*` 패키지중 하나를 선택해서 깔고 소스에서 자신에 맞게 만들어 써야한다.

커널을 업그레이드 하기 위해서. 여러분의 하위아키텍처에 맞는 것을 선택하고 모른 커널 목록은 다음 명령으로 찾을 수 있다:

```
# apt-cache search ^kernel-image
```

설치를 하기 위해서 `aptitude install`을 해야한다. 새로운 커널을 설치하게 되면 재부팅을 해야 동작하게 된다.

woody 설치시스템은 여러분 시스템에 커널을 설치하지 않을 것이다. 이는 sarge에서 변했고 여러 분은 버추얼 패키지를 설치해서 커널 변화를 계속 따라갈 수 있다. 이 패키지들은 kernel-image-VERSION-ARCH 으로 이름을 짓는데 VERSION은 커널버전을 뜻하는 2.4나 2.6을 뜻하고 ARCH는 지원 아키텍처를 뜻한다. 패키지 관리에 들어가 있는 커널에 대한 보안 지원을 원한다면 커널 패키지를 설치하는데 업그레이드 뒤에 여러분 하드웨어에 딱맞는 것을 쓰면 된다.

더 많은 모험을 좋아한다면, Debian GNU/Linux에서 여러분에 맞는 커널을 컴파일 하는 손쉬운 방법이 있다. `kernel-package` 도구를 설치하고 `/usr/share/doc/kernel-package`에 있는 문서를 읽으면 된다.

4.6 오래된 패키지들

수천개의 새로운 패키지가 sarge에 들어왔고 또 2천개 패키지가 예전 woody에 있던 오래된 것은 빠져나갔다. 더이상 업그레이드가 없어서 퇴출당했고 필요하면 이 오래된 패키지 를 써도 상관은 없는데 데비안 프로젝트는 대개 보안 지원을 sarge가 나온 뒤로 1년 동안 연속으로 하지 않을 것이다 [3], 그리고 그동안 다른 지원은 보통 하지 않을 것이다. 이 패키지들을 다른 대체할 수 있는 것으로 쓰기를 권하는 바이다.

왜 패키지가 배포본에서 없어질 수 밖에 없는지 많은 이유가 있다: 더이상 상위 소스 개발이 되지 않는다; 더이상 그 패키지 를 관리하는 데비안 관리자가 없다; 그 패키지들이 제공하는 기능이 다른(보통 새버전) 패키지가 대체했다; 또는 버그 때문에 sarge에 적합하지 않는 경우가 있다. 보통 마지막 경우는 "불안정" 배포본에 여전히 있게 된다.

업데이트 시스템에서 "오래된" 패키지를 알아내는 방법은 아주 쉬운데 패키지 관리 프론트-엔드들이 이를 쉽게 알려준다. `aptitude`를 쓴다면, 이들 패키지를 "Obsolete and Locally Created Packages" 항목에서 보여줄 것이다. `dselect`는 비슷한 선택을 보여주긴 하지만 그 목록은 다르다. `aptitude`를 써서 패키지를 woody 있던걸 수동으로 설치했다면, 이 패키지를 추적해줄 것이고 오래된 패키지라고 의존성에서 따로 떨어져서 있게 되는데 패키지를 지우려면 더이상 필요하지 않다고 알려줄 것이다. 또, `deborphan`과 달리, `aptitude`는 수동으로 설치한 오래된 패키지를 표시 하지 않을 것인데 이는 자동으로 의존성 때문에 걸린 놈들과는 반대로 작동한다.

오래된 패키지를 찾는 다른 방법은 `deborphan`, `debfoster`, `cruft`가 있다. `deborphan`을 가장 추천한다. 그런데 기본으로 오래된 라이브러리를 보여줄것인데: "libs"나 "oldlibs" 섹션에 있는 패키지들은 다른 패키지들이 쓰지 않는 것들이다. 무조건 눈가리고 여기에 나온 패키지를 지우지 마라 잘못된 정보를 줄 수 있는 경우가 종종 있기 때문이다. 패키지를 하나하나 찾아보고 지우는 것이 가장 좋다(내용과 크기, 설명).

Debian Bug Tracking System (http://bugs.debian.org/)은 가끔 왜 패키지들이 없어졌는지 이유를 알려준다. 모아진 버그 보고를 그 패키지 자체에 대해서 살펴보고 그 내용은 ftp.debian.org pseudo-package (http://bugs.debian.org//cgi-bin/pkgreport.cgi?pkg=ftp.debian.org\&archive=yes)를 살펴보기 바란다.

[3] 보통 이 시간동안 새로운 릴리즈는 나오지 않는다. 전통적으로 두가지 안정 릴리즈가 같은 시간에 지원을 받는다.

4.6.1 Dummy packages

woody에서 온 몇몇 패키지들은 sarge에서 나뉘어서 시스템 관리가 편하게 만들었다. 이런 경우 업그레이드 경로를 쉽게 다루기 위해 sarge는 “dummy” 패키지를 준다: 새로운 패키지를 설치하게 만드는 의존성을 가진 woody에 있는 오래된 패키지와 동일한 이를을 가진 비어있는 패키지들. 이 “dummy” 패키지들은 업그레이드 뒤에 오래된 패키지가 되고 쉽게 없앨 수 있게 된다.

다 그런건 아니지만 대부분dummy 패키지들의 설명을 보면 그것들의 목적을 알 수 있다. dummy 패키지에 있는 패키지 설명은 균일하진 않고 `deborphan`에서 `--guess` 옵션으 로 여러분의 시스템에서 검색할 수 있다. 몇몇 dummy 패키지들은 업그레이드 뒤에 없어지지 않게 되었는데 대신 프로그램의 가능한 현재 버전을 따라가게 만들었다.

Chapter 5

자세한 시스템 변화

5.1 파이썬 패키지에 대한 변화

sarge 있는 파이썬 2.X 패키지의 어떤 것도 'profile'과 'pstats' 표준 모듈을 포함하지 않는다. 왜냐하면 이들은 DFSG와 호환이 안되는 라이센스를 가지고 있기 때문이다(버그 #293932를 참조하면 자세한 내용을 알 수 있다). 이 두 모듈은 python-profiler와 python2.X-profiler 패키지로 찾을 수 있고 데비안 아카이브의 non-free에 들어있다.

Chapter 6

Debian GNU/Linux에 대한 더 많은 정보

6.1 더 읽을 거리

문서는 데비안 문서 프로젝트(DDP)에 있는데 여기서는 데비안 사용자와 개발자를 위해 수준높은 문서를 만드는게 목적이다. 문서는 데비안 지침, 데비안 새로운 관리자 안내서, 데비안 FAQ가 있고 더 많은 문서가 있다. 더 많은 목록은 DDP 웹 사이트 DDP website (`http://www.debian.org/doc/ddp`) 를 참조하라.

각 패키지에 대한 문서는 `/usr/share/doc/`*`package`* 에 있으며 여기에는 저작권 정보와 데비안에 특정된 내용과 상위 문서가 들어있다.

6.2 도움 구하기

도움을 주는 내용과 충고와 지원은 많지만 몬서를 잘보면 모든 해답이 있다는 것을 알 것이다. 여기서는 데비안을 새로 쓰고자 하는 사람들에게 간략한 소개를 하려고 한다.

6.2.1 메일링 리스트

데비안 사용자들이 관심을 가질만한 리스트는 debian-user 리스트와 다른 debian-user-*language* 리스트가 있다. 전자는 영어이고 후자는 각국 언어이다. 이 리스트에 대한 정보와 자세한 사항과 구독은 `http://lists.debian.org/` 를 참조하다. 여러분이 질문할 문제가 이미 나와있는지 우선 찾아보고 나서 질문을 올리는 예의는 지키면 좋겠다.

6.2.2 인터넷 릴레이 채팅

데비안은 IRC 채널을 이용해서 사용자들에게 도움을 주고 있다. 이 서버는 오픈 프로젝트 IRC 네트워크에 있는데 오픈소스 공동체 사이에 정보를 나누기 위해 만들어졌다. 여기에 접속하려면 여러 분이 좋아하는 irc 클라이언트로 irc.openprojects.net로 접속해서 join #debian으로 들어가라.

채널 지침을 잘 따르고 상대방을 존중하면 된다. 자세한 내용은 website (http://www.openprojects.net/)를 참조하라.

6.3 버그 보고하기

우리는 데비안 GNU/리눅스를 최상의 품질의 OS로 만들고 있으나 모든 패키지가 버그가 없는 것은 아니다. 사용자들에게 이런 서비스를 주고자 버그에 대한 모든 정보를 우리의 버그추적시스템 (BTS)에서 보여주는데 bugs.debian.org (http://bugs.debian.org/) 서 확인가능하며 이는 데비안의 "열린 개발"정책과 일치한다.

패키지에 속해있거나 배포본에 있는 버그를 발견했다면 이를 보고해서 다음 릴리즈에 문제가 없게 만들면 된다. 버그보고에는 이메일 주소가 필요하고 버그를 추적하고 개발자가 보고자와 이야기를 잘 해서 더 많은 정보를 얻게 하면 된다.

`reportbug`을 써서 버그 보고를 하거나 하나하나 전자우편으로 보낼 수 있다. 자세한 사항은 버그 추적 시스템 을 읽고 참조 카드(`/usr/share/doc/debian`)를 읽거나 Bug Tracking System (http://bugs.debian.org/)보고 할 수 있다.

6.4 데비안에 도움주기

데비안에 도움을 주는데 전문가가 될 필요는 없다. 여러가지 상황에서 문제를 가진 다른 사용자를 도와주면서 (lists (http://lists.debian.org/) 공동체에 도움을 줄 수 있다. 개발과 관련된 lists (http://lists.debian.org/)에 문제를 확인하고 해결책을 주는 일은 매우 좋은 일이다. 데비안을 높은 수준으로 유지 하기 위해서 버그 보고 (http://bugs.debian.org/)를 통해서 개발자들이 추적을 하고 수정하게 된다. 데비안 프로젝트에 좀더 많은 도움을 적즉적으로 주려던 문서 (http://www.debian.org/doc/ddp)를 쓰거나 기존에 있는 문서를 자신의 언어로 id="http://www.debian.org/international/" name="번역"> 하는 일이다.

좀더 시간을 내서 하겠다면, 데비안 안에서 자유소프트웨어 모음을 관리할 수 있다. 데비안 안에서 패키지 요청을 하는 패키지를 관리하거나 입양을 하는 일이 도움이 될 것인데 이는 Work Needing and Prospective Packages database (http://www.debian.org/devel/wnpp/)에서 자세하게 알아 볼 수 있을 것이다. 데비안 포팅을 포함한 데비안 안에서 재밌게 할 수 있는 부분이 있다면, Debian Jr. (http://www.debian.org/devel/debian-jr/)와 Debian Med (http://www.debian.org/devel/debian-med/)도 있으니 참고 바란다.

어떤 경우라도, 사용자이건, 프로그래머건, 문서 작성을 하건 번역을 하건 자유 소프트웨어 공동체에서 일하고 있다면 이미 자유 소프트웨어 에 도움을 주고 있는 것이다. 공동체에 도움을 주는 일은 다시 보답을 받는 일이면서 즐겁고 새로운 사람들을 만나면서 따뜻한 마음을 갖게 되는 일이다.

Appendix A

woody 시스템 관리하기

이 부록은 패키지를 설치하거나 woody 패키지에서 업그레 이드 하는 방법을 담고 있는데 항상 sarge로 업그레이드 하기 전에 확인하라. 특정의 경우에 해당한다는 것을 알아둬라.

A.1 woody 시스템 업그레이드하기

기본적으로 woody에서 업그레이드 했던 것과 크게 다르지 않다. 단지 차이라면 여러분 패키지 목록에 '소스 리스트 확인' on this page에서 설명한 woody 패키지들이 있다는 점이다.

A.2 aptitude의 woody 버전 설치

우선 woody의 `aptitude`를 설치할 것이고 sarge의 `aptitude` 아리라는 것을 확인하라. 이 내용은 '소스 리스트 확인' on the current page 설명하였다.

이 뒤로, 다음을 실행하라.

```
# apt-get install aptitude
```

그리하여 `aptitude` 설치하라.

A.3 소스 리스트 확인

만일 `/etc/apt/sources.list`에서 'stable'을 쓰고 있다면 이미 sarge를 "사용하고" 있는 중이다. 이미 `apt-get update` 했다면, 다음 사항을 문제 없이 따른다.

이미 sarge 패키지를 쓰고 있다면, 더이상 woody 에서 패키지를 쓸 수 없다. 이런 경우 여러분 스스로 계속 해 나갈 것인지 말 것인지 결정해야한다. 패키지를 다운 그레이드 할 수도 있지만 여기선 다루지 않는다.

`/etc/apt/sources.list`을 아무 편집기나 써서 root로 열어서 `deb http:`나 `deb ftp:` 부분이 "`stable`" 을 참조하는지 확인하라. 그런 부분이 있으면, `stable`을 `woody`로 바꾸라.

`deb file:`로 시작하는 부분이 있으면 직접 확인해서 위치가 woody인지 sarge인지 어떤 부분을 리스트가 참조하는지 확인하라.

중요! `deb cdrom:`으로 시작하는 부분은 바꾸지 마라. 이 부분을 바꾸면 `apt-cdrom` 실행해서 다시 복구 해야 한다. 'cdrom'부분이 "`unstable`"를 참조하는 것에 놀라지 마라. 혼란이 올지 모르지만 정상이다.

변화를 줬다면 파일을 저장하고 다음을 실행하라.

```
# apt-get update
```

이러면 패키지 리스트를 새롭게 할 수 있다.