데비안 GNU/리눅스 설치 안내서

2022년 3월 27일

데비안 GNU/리눅스 설치 안내서

이 설명서의 빌드 버전: 20220129~deb11u1.

차 례

1 데비안에 오신 것을 환영합니다 **1**

1.1 데비안이란? . . . 1
1.2 GNU/리눅스란? . . . 2
1.3 데비안 GNU/리눅스란? . . . 3
1.4 데비안 설치 프로그램이란? . . . 3
1.5 데비안 받기 . . . 4
1.6 이 문서의 최신 버전 구하는 법 . . . 4
1.7 이 문서의 구성 . . . 4
1.8 저작권 및 소프트웨어 라이선스 정보 . . . 5

2 시스템 요구 사항 **7**

2.1 지원하는 하드웨어 . . . 7
2.1.1 지원하는 아키텍처 . . . 7
2.1.2 3가지 ARM 포트 . . . 8
2.1.3 ARM CPU 설계의 다양함과 복잡한 지원 . . . 8
2.1.4 데비안/armhf가 지원하는 플랫폼 . . . 9
2.1.5 다중 프로세서 . . . 11
2.1.6 그래픽 하드웨어 지원 . . . 11
2.1.7 네트워크 연결 하드웨어 . . . 11
2.1.8 주변 장치 및 기타 하드웨어 . . . 11
2.2 펌웨어가 필요한 장치 . . . 11
2.3 GNU/Linux에 적합한 하드웨어 구입 . . . 12
2.3.1 독점적이거나 폐쇄된 하드웨어 피하기 . . . 12
2.4 설치 미디어 . . . 13
2.4.1 CD-ROM/DVD-ROM/BD-ROM . . . 13
2.4.2 네트워크 . . . 13
2.4.3 하드디스크 . . . 13
2.4.4 유닉스 계열 혹은 GNU 시스템 . . . 13
2.4.5 지원하는 저장 장치 . . . 14
2.5 메모리 및 디스크 공간 요구 사항 . . . 14

3 데비안 GNU/리눅스를 설치하기 전에 **15**

3.1 설치 과정 개요 . . . 15
3.2 기존 데이터를 백업하십시오! . . . 16
3.3 필요한 정보 . . . 17
3.3.1 문서 . . . 17
3.3.1.1 설치 안내서 . . . 17
3.3.1.2 하드웨어 문서 . . . 17

3.3.2 하드웨어 정보가 있는 곳 찾기 . . . 17
3.3.3 하드웨어 호환성 . . . 18
3.3.3.1 라이브 시스템에서 하드웨어 호환성 검사하기 . . . 18
3.3.4 네트워크 설정 . . . 19
3.4 최소 하드웨어 요구 사항 맞추기 . . . 19
3.5 멀티 부팅 시스템에서 미리 파티션하기 . . . 20
3.6 설치하기 전에 할 하드웨어 및 운영 체제 설정 . . . 21
3.6.1 ARM 펌웨어 . . . 21
3.6.2 데비안 공급 U-Boot (시스템 펌웨어) 이미지 . . . 21
3.6.3 U-Boot에서 이더넷 MAC 주소 설정하기 . . . 21
3.6.4 U-Boot의 커널/최초 램디스크/디바이스 트리 재배치 문제 . . . 22

4 시스템 설치 미디어 구하기 23
4.1 공식 데비안 GNU/리눅스 설치 이미지 . . . 23
4.2 데비안 미러에서 파일 다운로드 . . . 23
4.2.1 설치 파일을 찾을 위치 . . . 24
4.2.1.1 armhf 멀티플랫폼 설치 파일 . . . 24
4.3 TFTP 네트워크 부팅에 필요한 파일 준비하기 . . . 24
4.3.1 RARP 서버 준비하기 . . . 24
4.3.2 DHCP 서버 준비하기 . . . 24
4.3.3 BOOTP 서버 준비하기 . . . 25
4.3.4 TFTP 서버 사용하기 . . . 26
4.3.5 TFTP 이미지를 적당한 위치에 놓기 . . . 26
4.4 자동 설치 . . . 26
4.4.1 데비안 설치 프로그램을 이용한 자동 설치 . . . 27
4.5 설치 파일의 무결성 확인하기 . . . 27

5 설치 시스템 부팅하기 28
5.1 32-bit hard-float ARMv7에서 설치 프로그램 부팅하기 . . . 28
5.1.1 부팅 이미지 형식 . . . 28
5.1.2 콘솔 설정 . . . 28
5.1.3 TFTP로 부팅하기 . . . 28
5.1.3.1 U-Boot에서 TFTP 부팅 . . . 29
5.1.3.2 미리 빌드된 네트워크 부팅 타르볼 . . . 30
5.1.4 U-Boot 이용해 USB 메모리에서 부팅하기 . . . 30
5.1.5 설치 프로그램에서 빌드한 SD 카드 이미지 사용하기 . . . 31
5.2 접근성 . . . 31
5.2.1 설치 프로그램 프론트엔드 . . . 32
5.2.2 보드 장치 . . . 32
5.2.3 고대비 테마 . . . 32
5.2.4 화면 확대 . . . 32
5.2.5 전문가 설치, 복구 모드, 자동 설치 . . . 32
5.2.6 설치한 시스템의 접근성 . . . 33

5.3 부팅 파라미터 . . . 33
5.3.1 부팅 콘솔 . . . 33
5.3.2 데비안 설치프로그램 파라미터 . . . 33
5.3.3 부팅 파라미터로 질문에 답하기 . . . 36
5.3.4 커널 모듈에 파라미터 넘기기 . . . 36
5.3.5 커널 모듈 블랙리스트 . . . 37
5.4 설치 과정의 문제 해결 . . . 37
5.4.1 광학 미디어의 안정성 . . . 37
5.4.1.1 공통 사항 . . . 38
5.4.1.2 문제점 파악 및 해결 방법 . . . 38
5.4.2 부팅 설정 . . . 39
5.4.3 커널 시작 메시지 해석하기 . . . 39
5.4.4 설치 문제 보고하기 . . . 39
5.4.5 설치 보고 제출 . . . 40

6 데비안 설치 프로그램 사용하기 42
6.1 설치 프로그램이 동작하는 방식 . . . 42
6.2 컴포넌트 소개 . . . 43
6.3 컴포넌트 사용하기 . . . 44
6.3.1 데비안 설치 프로그램 준비 및 하드웨어 설정 . . . 45
6.3.1.1 사용 가능 메모리 검사 / 저용량 메모리 모드 . . . 45
6.3.1.2 지역화 옵션 선택 . . . 45
6.3.1.3 키보드 선택하기 . . . 46
6.3.1.4 데비안 설치 프로그램 ISO 이미지 찾기 . . . 46
6.3.1.5 네트워크 설정하기 . . . 47
6.3.1.5.1 네트워크 자동 설정 . . . 47
6.3.1.5.2 네트워크 수동 설정 . . . 47
6.3.1.5.3 IPv4 및 IPv6 . . . 48
6.3.2 사용자 및 암호 설정 . . . 48
6.3.2.1 루트 암호 설정 . . . 48
6.3.2.2 일반 사용자 만들기 . . . 48
6.3.3 시계 및 시간 설정 . . . 49
6.3.4 파티션하기 및 마운트 위치 선택 . . . 49
6.3.4.1 지원하는 파티션 옵션 . . . 49
6.3.4.2 자동 파티션하기 . . . 50
6.3.4.3 수동 파티션하기 . . . 52
6.3.4.4 멀티디스크 장치 설정하기 (소프트웨어 RAID) . . . 53
6.3.4.5 LVM (논리 볼륨 관리자) 설정하기 . . . 55
6.3.4.6 암호화 볼륨 설정하기 . . . 56
6.3.5 베이스 시스템 설치하기 . . . 58
6.3.6 추가 소프트웨어 설치하기 . . . 59
6.3.6.1 APT 설정하기 . . . 59
6.3.6.1.1 여러 개의 CD나 DVD 이미지에서 설치하기 . . . 60

6.3.6.1.2 네트워크 미러 사용하기 60
6.3.6.1.3 네트워크 미러 고르기 61
6.3.6.2 소프트웨어 선택 및 설치 61
6.3.7 시스템을 부팅 가능하게 만들기 62
6.3.7.1 다른 운영 체제 검색 63
6.3.7.2 시스템을 flash-kernel로 부팅 가능하게 만들기 63
6.3.7.3 부트로더 없이 계속 63
6.3.8 설치 마치기 63
6.3.8.1 시스템 시계 맞추기 63
6.3.8.2 시스템 다시 시작 64
6.3.9 문제해결 64
6.3.9.1 설치 로그 저장 64
6.3.9.2 쉘 사용하기 및 로그 보기 64
6.3.10 네트워크 콘솔을 통해 설치 65
6.4 없는 펌웨어 읽어들이기 66
6.4.1 미디어 준비하기 67
6.4.2 펌웨어 및 설치한 시스템 67
6.4.3 시스템 설치 마치기 68
6.5 원하는대로 바꾸기 69
6.5.1 다른 init 시스템 설치하기 69

7 새로운 데비안 시스템으로 부팅하기 70
7.1 진실의 시간 70
7.2 암호화 볼륨 마운트하기 70
7.2.1 문제 해결 71
7.3 로그인 71

8 다음 단계 및 그 다음에 할 일 73
8.1 시스템 끄기 73
8.2 데비안에 익숙해지기 73
8.2.1 데비안 패키지 시스템 73
8.2.2 데비안용 추가 소프트웨어 74
8.2.3 프로그램 버전 관리 74
8.2.4 CRON 작업 관리 74
8.3 그 외의 읽을 거리 및 정보 74
8.4 시스템에 전자메일 준비하기 75
8.4.1 기본 전자메일 설정 75
8.4.2 시스템 외부에 전자메일 보내기 76
8.4.3 Exim4 MTA 설정하기 76
8.5 새 커널 컴파일하기 77
8.6 손상된 시스템 복구하기 77

A 설치 방법 **78**
A.1 들어가기 전에 . . . 78
A.2 설치 프로그램 시작하기 . . . 78
A.2.1 광학 디스크 . . . 78
A.2.2 네트워크 부팅 . . . 78
A.2.3 하드 디스크 부팅 . . . 79
A.3 설치 . . . 79
A.4 설치 보고서를 보내주십시오 . . . 80
A.5 그리고 마지막으로... . . . 80

B 미리 설정을 이용한 설치 자동화 **81**
B.1 소개 . . . 81
B.1.1 미리 설정 방법 . . . 81
B.1.2 한계 . . . 82
B.2 미리 설정 사용하기 . . . 82
B.2.1 미리 설정 파일 읽어들이기 . . . 82
B.2.2 부팅 파라미터로 미리 설정하기 . . . 83
B.2.3 자동 모드 . . . 84
B.2.4 미리 설정할 때 쓸모 있는 줄임말 . . . 85
B.2.5 부팅 프롬프트 미리 설정 예제 . . . 86
B.2.6 미리 설정 파일을 지정하는 데 DHCP 서버 사용하기 . . . 86
B.3 미리 설정 파일 만들기 . . . 87
B.4 미리 설정 파일의 내용 (bullseye용) . . . 88
B.4.1 지역화 . . . 88
B.4.2 네트워크 설정 . . . 89
B.4.3 네트워크 콘솔 . . . 91
B.4.4 미러 사이트 설정 . . . 91
B.4.5 계정 설정 . . . 92
B.4.6 시계 및 시간대 설정 . . . 93
B.4.7 파티션하기 . . . 93
B.4.7.1 파티션 예제 . . . 93
B.4.7.2 RAID를 사용해 파티션하기 . . . 96
B.4.7.3 파티션 마운트 방법 조정하기 . . . 97
B.4.8 기본 시스템 설치 . . . 98
B.4.9 APT 설정 . . . 98
B.4.10 패키지 선택 . . . 99
B.4.11 설치 마치기 . . . 100
B.4.12 기타 패키지 미리 설정 . . . 100
B.5 고급 옵션 . . . 101
B.5.1 설치할 때 임의의 명령어 실행하기 . . . 101
B.5.2 미리 설정을 이용해 기본값 바꾸기 . . . 101
B.5.3 미리 설정 파일을 분리해서 사용하기 . . . 102

C 데비안에서 파티션 나누기 **103**
C.1 데비안 파티션 및 크기 정하기 . . . 103
C.2 디렉터리 구조 . . . 103
C.3 권장하는 파티션 구조 . . . 105
C.4 리눅스의 장치 이름 . . . 105
C.5 데비안의 파티션 프로그램 . . . 106

D 여러가지 내용 **107**
D.1 리눅스 장치 . . . 107
D.1.1 마우스 설정하기 . . . 108
D.2 태스크마다 필요한 디스크 공간 . . . 108
D.3 유닉스/리눅스 시스템에서 데비안 GNU/리눅스 설치하기 . . . 109
D.3.1 시작하기 . . . 109
D.3.2 **debootstrap** 설치 . . . 110
D.3.3 **debootstrap** 실행 . . . 111
D.3.4 베이스 시스템 설정 . . . 111
D.3.4.1 장치 파일 만들기 . . . 111
D.3.4.2 파티션 마운트하기 . . . 112
D.3.4.3 시간대 설정하기 . . . 113
D.3.4.4 네트워크 설정하기 . . . 113
D.3.4.5 APT 설정하기 . . . 114
D.3.4.6 로캘 및 키보드 설정하기 . . . 114
D.3.5 커널 설치 . . . 115
D.3.6 부트로더 설정하기 . . . 115
D.3.7 원격 접근: SSH 설치 및 접근 설정 . . . 115
D.3.8 마지막 처리 . . . 116
D.4 PPP 오버 이더넷을(PPPoE) 이용해 데비안 GNU/리눅스 설치하기 . . . 116

E 문서 관리 정보 **118**
E.1 문서 정보 . . . 118
E.2 이 문서에 참여하기 . . . 118
E.3 중요 기여자들 . . . 119
E.4 상표권 안내 . . . 119

F GNU 일반 공중 사용 허가서 **120**
F.1 전문 . . . 120
F.2 GNU 일반 공중 사용 허가서 (GNU GENERAL PUBLIC LICENSE) . . . 121
F.3 새로운 프로그램에 GPL을 적용하는 방법 . . . 125

표 차례

3 데비안 GNU/리눅스를 설치하기 전에

3.1 설치에 도움이 되는 하드웨어 정보 17

3.2 추천하는 최소 시스템 요구사항 19

요 약

이 문서는 32-bit hard-float ARMv7(“armhf”) 아키텍처 데비안 GNU/리눅스 11 시스템 (코드명 “bullseye”) 설치 설명서입니다. 또 이 문서에는 자세한 정보를 찾을 수 있는 곳을 안내합니다. 또 새로 설치한 데비안 시스템을 잘 활용하는 방법도 들어 있습니다.

이 한국어 번역은 여러 분들이 자발적으로 참여해 완성되었습니다. 번역한 내용 중에 어색한 부분이나 잘못 번역된 부분이 있다면 데비안 한국어 지역화 메일링 리스트에 (debian-l10n-korean@lists.debian.org) 메일을 보내주시면 반영하겠습니다.

armhf용 데비안 GNU/리눅스 11 설치하기

데비안을 사용해 주셔서 기쁩니다. 데비안을 사용하게 되시면 데비안의 GNU/Linux 배포판이 유일무이하다는 걸 알게 되실 겁니다. 데비안 GNU/리눅스는 전세계의 고품질의 자유 소프트웨어를 모은 배포판으로, 일관적으로 통합되어 있습니다. 저희는 이렇게 모아 놓은 배포판이 개별 자유 소프트웨어 이상의 힘을 발휘한다고 생각합니다.

많은 분들은 이 설명서를 읽지 않고 데비안을 설치하려 할 겁니다. 실제로 데비안 설치 프로그램은 설명서를 보지 않고도 설치할 수 있도록 설계되어 있습니다. 설치 안내서를 모두 읽을 시간이 없으시면, 설치 하우투(Howto)를 읽어 보시는 걸 추천합니다. 설치 하우투에는 기본적인 설치 과정이 설명되어 있고, 고급 정보나 문제가 생겼을 경우에 대한 설명서 링크가 들어 있습니다. 설치 하우투는 부록 A에 있습니다.

그렇긴 하지만 시간을 내서 이 안내서 대부분을 읽어보시기를 바랍니다. 그러면 더 많이 알고 있는 상태에서 아마도 더 성공적으로 설치할 수 있을 겁니다.

제 1 장

데비안에 오신 것을 환영합니다

이 장에서는 데비안 프로젝트와 데비안 GNU/리눅스에 대해 간단히 설명합니다. 데비안 프로젝트의 역사와 데비안 GNU/리눅스 배포판에 대해 이미 알고 있다면 다음 장으로 넘어가셔도 됩니다.

1.1 데비안이란?

데비안은 자유 소프트웨어를 개발하고 자유 소프트웨어 커뮤니티의 이상을 널리 알리는 일을 위해 조직한, 자원자로만 구성된 조직입니다. 데비안 프로젝트는 1993년에 이안 머독이 새로운 (당시에는 비교적 새로운 버전의) 커널을 사용하면서, 완전하고 일관된 소프트웨어 배포판을 만드는 데 참여할 소프트웨어 개발자를 공개적으로 모집하면서부터 시작되었습니다. 처음에는 자유 소프트웨어 재단(Free Software Foundation)의 자금 지원을 받았고, GNU 철학에 영향을 받은 비교적 작은 규모의 이 열성적인 집단은, 해를 거듭하면서 1000여 명의 데비안 개발자가 참여하는 조직으로 성장했습니다.

데비안 개발자는 홈페이지 및 FTP 사이트 관리, 그래픽 디자인, 소프트웨어 라이선스의 법률적 해석, 문서 작성, (말할 필요도 없이) 소프트웨어 패키지 관리 등 다양한 활동에 참여합니다.

데비안의 철학을 전달하고, 또 데비안의 원칙에 동의하는 개발자를 끌어 모으려고 데비안 프로젝트는 여러가지 문서를 발표했습니다. 이러한 문서에서 데비안의 가치를 간략히 설명합니다. 또 데비안 개발자가 되는 게 무엇을 뜻하는 지에 대한 지침 역할을 합니다:

- 데비안 우리의 약속(Debian Social Contract)은 자유소프트웨어 공동체에 대한 데비안의 선언입니다. 이 선언에 따르기로 동의한 사람은 누구나 메인테이너가 될 수 있습니다. 모든 메인테이너는 새로운 소프트웨어를 데비안에 추가할 수 있습니다. 단 그 소프트웨어가 우리의 자유소프트웨어 기준에 맞아야 하고, 패키지가 우리의 품질 기준에 맞아야 합니다.

- 데비안 자유 소프트웨어 기준(Debian Free Software Guidelines)은 자유소프트웨어에 대한 데비안의 기준을 단순명료하게 설명한 것입니다. DFSG는 자유소프트웨어 운동에 있어서 매우 영향력 있는 문서로, 이 문서를 기초로 오픈 소스 정의(Open Source Definition)가 작성되었습니다.

- 데비안 정책 설명서(Debian Policy Manual)는 데비안 프로젝트의 품질 기준을 자세히 적은 명세서입니다.

데비안 개발자는 여러가지 다른 프로젝트에 참가하기도 합니다. 데비안과 관계된 프로젝트도 있고, 리눅스 공동체의 일부 혹은 전부와 관련되어 있는 프로젝트도 있습니다. 예를 들면 다음과 같습니다:

- Filesystem Hierarchy Standard (FHS) 프로젝트는 리눅스 파일 시스템의 구성을 표준화하는 프로젝트입니다. FHS의 표준화를 통해 소프트웨어 개발자는 개발한 패키지가 GNU/리눅스 배포판에 따라 어떻게 설치되는 지 고민할 필요 없이, 프로그램 설계에만 전념할 수 있습니다.
- 데비안 쥬니어 (Debian Junior) 프로젝트는 데비안 내부 프로젝트로 어린이 사용자가 사용할 만한 데비안을 만드는 프로젝트입니다.

데비안에 관해 더 일반적인 정보는 데비안 FAQ를 참고하십시오.

1.2 GNU/리눅스란?

GNU/리눅스는 운영체제입니다. 운영체제는 여러가지 프로그램의 모음으로, 이 프로그램을 이용해 컴퓨터를 사용하고 다른 프로그램을 실행하기도 합니다.

운영체제는 컴퓨터에 필요한 여러 가지 기초적인 프로그램으로 구성되어 있고, 이 프로그램을 이용해 사용자는 컴퓨터와 의사소통을 하고 컴퓨터에 지시를 내립니다. 예를 들어 하드 디스크, 테이프, 프린터로 데이터를 보내거나 여기에서 데이터를 읽어들이며, 메모리 사용을 제어하고, 다른 프로그램을 실행합니다. 운영체제의 가장 중요한 부분이 바로 커널입니다, GNU/리눅스 시스템에서 리눅스는 커널 부분을 말합니다. 시스템의 나머지 부분은 기타 프로그램으로 구성되며, GNU 프로젝트가 많은 부분을 개발했습니다. 리눅스 커널 그 자체만으로는 시스템을 구성할 수 없기 때문에, 우리는 흔히 리눅스라고 호칭하는 시스템을 GNU/리눅스라는 이름으로 사용합니다.

GNU/리눅스는 유닉스(Unix) 운영체제를 모델로 만든 운영체제입니다. 애초부터 GNU/리눅스는 다중 작업, 다중 사용자 시스템으로 설계되었습니다. 다중 작업, 다중 사용자 시스템이라는 것만으로도 리눅스는 여타의 잘 알려진 운영체제와 차별됩니다. 그러나 GNU/리눅스는 여러분이 생각하는 것 이상으로 다른 점이 아주 많습니다. 다른 운영체제와는 달리 어느 누구도 GNU/리눅스를 소유하지 않습니다. 자원자의 힘으로 GNU/리눅스의 상당 부분을 개발했습니다.

후에 GNU/리눅스라고 불리게 된 시스템의 개발은 1984년에 시작되었으며, 이 때 자유 소프트웨어 재단 (FSF)은 유닉스와 유사한 운영체제의 개발을 시작하면서 그 이름을 GNU라고 했습니다.

GNU 프로젝트는 유닉스(Unix™) 및 GNU/리눅스처럼 유닉스와 유사한 운영체제에서 사용할 수 있는 일련의 자유 소프트웨어 도구를 개발해 왔습니다. 이러한 도구를 이용해 사용자는 파일을 복사하거나 지우는 아주 일상적인 작업부터, 프로그램 작성과 컴파일, 여러가지 종류의 문서 편집에 이르기까지 다양한 작업을 할 수 있습니다.

수많은 단체와 개인이 GNU/리눅스의 개발에 기여했지만, 단독으로 가장 크게 기여한 곳은 자유 소프트웨어 재단(Free Software Foundation)입니다. 자유 소프트웨어 재단은 GNU/리눅스에서 사용하는 도구의 대부분을 개발했을 뿐 아니라, GNU/리눅스가 생겨나게 했던 철학과 공동체를 만들어 냈습니다.

리눅스 커널은 리누스 토발즈(Linus Torvalds)라는 핀란드의 한 컴퓨터 과학 대학생이 1991년 유즈넷 뉴스그룹 **comp.os.minix**에 미닉스(Minix)를 대체하는 커널의 초기 버전을 발표하면서 처음으로 그 모습을 세상에 드러냈습니다. 자세한 사항은 리눅스 인터네셔널의 리눅스 역사 페이지(Linux History Page)를 참고하십시오.

리누스 토발즈는 몇 명의 서브시스템 관리자의 도움을 받아 수백명에 달하는 개발자의 작업을 조율하고 있습니다. 리눅스 커널의 공식 홈페이지가 있습니다. **linux-kernel** 메일링 리스트에 대한 정보는 linux-kernel 메일링 리스트 FAQ에서 찾아 보십시오.

GNU/리눅스 사용자는 소프트웨어 선택에 많은 자유를 갖고 있습니다. 예를 들어 GNU/리눅스 사용자는 열 개가 넘는 종류의 다른 커맨드 라인 쉘, 여러가지 그래픽 데스크톱 중에서 원하는 소프트웨어를 선택할 수

있습니다. 이와 같이 선택의 폭이 넓어서 쉘이나 데스크톱을 바꿀 수 있다는 걸 전혀 생각치 못했던 사용자가 당황스러워 하기도 합니다.

또한 GNU/리눅스는 여타 운영체제에 비해 시스템이 멈추는 경우가 적고, 동시에 둘 이상의 프로그램을 실행하는 성능이 월등하고, 보안에 강합니다. 리눅스는 서버 시장에서 가장 성장이 빠른 운영체제입니다. 최근에 리눅스는 가정과 업무용 사용자에게도 퍼져 나가고 있습니다.

1.3 데비안 GNU/리눅스란?

데비안의 철학 및 방법론과 GNU 도구, 리눅스 커널, 그리고 기타 중요한 자유소프트웨어가 모여 데비안 GNU/리눅스라는 독특한 배포판을 만듭니다. 이 배포판은 수많은 소프트웨어 패키지로 구성됩니다. 배포판의 각 패키지는 실행파일, 스크립트, 문서, 설정 정보가 들어 있으며 메인테이너가 관리합니다. 메인테이너는 각 패키지를 항상 최신으로 유지하고, 버그 리포트를 추적하고, 패키지로 만든 소프트웨어의 원 개발자와 연락을 하고 있습니다. 데비안의 거대한 사용자 기반이 버그 추적 시스템과 결합해 문제점을 빠르게 찾아내고 수정할 수 있습니다.

데비안이 세세한 신경을 쓰는 덕분에 품질 높고, 안정적이고, 확장성 좋은 배포판을 만들어 냅니다. 간단한 설치 설정에 따라 방화벽에서 데스크톱 공학용 워크스테이션, 고성능 네트워크 서버까지 다양한 역할을 할 수 있습니다.

기술적으로 우수하면서 리눅스 공동체의 필요와 기대에 대해 적극적으로 참여하기 때문에, 데비안은 고급 사용자에게 특히 인기가 있습니다. 또한 데비안은 지금은 리눅스에 일반적인 기능이 된 많은 기능을 새로 도입하는 데 앞장 서 왔습니다.

예를 들어보면, 소프트웨어의 설치와 제거를 손쉽게 할 수 있는 패키지 관리 시스템이 들어간 최초의 리눅스 배포판이 데비안입니다. 또 다시 설치하지 않고도 업그레이드할 수 있는 최초의 배포판입니다.

데비안은 리눅스 개발의 리더로서 계속하고 있습니다. 그 개발 과정(전체 운영 체제를 구축하고 유지하는 매우 복잡한 작업하더라도)은 오픈 소스 개발 모델이 얼마나 잘 진행되는지를 보여주는되어 있습니다.

데비안은 다른 GNU/Linux 배포판과 다른 가장 큰 특징은 패키지 관리 시스템입니다. 데비안 시스템 관리자는 시스템에 설치된 패키지에 대해 하나의 패키지 설치에서 운영 체제 전체 자동 업데이트까지 완벽하게 제어할 수 있습니다. 개별 패키지를 업데이트하지 않도록 설정할 수 있습니다. 직접 컴파일한 소프트웨어에 대한 의존성을 설정할 수 있습니다.

“트로이 목마” 및 기타 악의적인 소프트웨어로부터 여러분의 시스템을 보호하려고 데비안 서버에서 업로드한 패키지가 등록된 데비안 개발자가 만든 패키지가 맞는지 여부를 확인합니다. 또한 데비안 각 패키지는 더 안전하게 설정되도록 세심한 주의를 기울입니다. 배포한 패키지에 보안 문제가 발생하면, 수정 버전을 빠르게 사용할 수 있습니다. 데비안의 간단한 업데이트 옵션을 사용해, 보안 패치를 인터넷에서 자동으로 다운로드하고 설치할 수 있습니다.

여러분의 데비안 GNU / Linux 시스템에 대한 지원을 받거나 데비안 개발자와 연락을 하는 가장 좋은 방법은 데비안 프로젝트에서 운영하는 여러가지 메일링 리스트를 사용하는 것입니다. (이 글이 작성된 시점에서 322개 이상의 메일링 리스트가 있습니다.) 메일링 리스트에 가입하려면, 데비안 메일링리스트 가입 페이지를 방문해 페이지의 양식을 기입하면 됩니다.

1.4 데비안 설치 프로그램이란?

데비안 설치 프로그램은 (Debian Installer, 줄여서 “d-i”라고도 합니다) 기본적인 동작을 하는 데비안 시스템을 설치하는 소프트웨어입니다. 임베디드 장치, 노트북 컴퓨터, 데스크톱 컴퓨터, 서버 컴퓨터 등 여러가지

종류의 하드웨어를 지원하고, 여러가지 목적으로 사용할 수 있는 커다란 자유 소프트웨어 모음을 제공합니다.

설치 과정은 기본적인 질문 모음에 답하는 것으로 진행됩니다. 또 전문가 모드에서는 설치 과정을 모두 제어하고 자동으로 설치할 수 있는 기능이 들어 있습니다. 설치한 시스템을 그대로 사용할 수도 있고 나중에 원하는대로 바꿀 수 있습니다. 설치는 USB, CD/DVD/Blu-Ray, 네트워크 등 여러가지 방식 중 하나를 이용할 수 있습니다. 설치 프로그램은 80개 이상의 언어를 통한 설치를 지원합니다.

설치 프로그램은 과거 boot-floppies 프로젝트에 기원을 두고 있고, 2000년 Joey Hess가 처음 언급했습니다. 그 이후로 자발적인 개발자들이 설치 시스템을 계속 발전시키고 새로운 기능을 추가해 왔습니다.

더 자세한 정보는 데비안 설치 프로그램 페이지, 위키, debian-boot 메일링 리스트에 있습니다.

1.5 데비안 받기

인터넷을 통해 데비안 GNU/리눅스를 다운로드하는 방법이나 데비안 공식 설치 미디어 구입에 대한 정보는 배포판 홈페이지를 참조하십시오. 데비안 미러 목록에는 데비안 공식 미러 사이트가 모두 정리되어 있기 때문에 가장 가까운 미러 사이트를 쉽게 찾을 수 있습니다.

데비안은 설치 후 아주 쉽게 업그레이드 할 수 있습니다. 설치 절차에 따라 일단 시스템 설치를 마치면 필요에 따라 업그레이드를 수행할 수 있습니다.

1.6 이 문서의 최신 버전 구하는 법

이 문서는 계속해서 개선됩니다. 데비안 GNU/리눅스 시스템의 11 릴리스에 대한 최신 정보는 데비안 11 페이지를 확인하시기 바랍니다. 이 설치문서의 최신 버전은 공식 데비안 설치 안내서 페이지에서도 보실 수 있습니다.

1.7 이 문서의 구성

이 문서는 처음으로 데비안을 사용하는 분을 위해 작성된 문서입니다. 가능한 한 전문적인 지식 없이도 읽을 수 있도록 노력하고 있습니다. 하지만 컴퓨터가 어떻게 동작하는지 일반적인 지식은 있다고 가정합니다.

전문적인 사용자도 이 문서에서 최소 설치에 필요한 공간과 데비안 설치 시스템에서 지원하는 하드웨어 등과 같이 도움이 될만한 정보를 얻을 수 있습니다. 숙련된 사용자는 이 문서에서 필요한 부분만 읽어도 됩니다.

이 안내서는 설치 과정의 처음부터 끝까지, 각 과정의 순서대로 쓰여 있습니다. 아래는 데비안 GNU/리눅스를 설치하는 과정으로, 각 과정에 관련된 문서 부분을 안내해 놓았습니다:

1. 2장에서, 하드웨어가 설치 시스템에 필요한 요구사항을 만족하는 지 판단합니다.
2. 3장에서는, 기존 시스템을 백업하고 데비안 설치에 앞서 필요한 계획을 하고 하드웨어 설정을 합니다. 멀티 부팅을 생각하고 계시다면, 하드 디스크에 데비안 파티션을 만들기 위한 공간을 만들어야 할 수도 있습니다.
3. 4장에서, 설치하는 방법에 따라 필요한 설치 파일을 구합니다.
4. 그 다음 5장에서는, 설치 시스템으로 부팅하는 방법을 설명합니다. 이 과정에서 문제가 발생한 경우 문제를 해결하는 방법도 이 장에서 설명합니다.

5. 6장에 따라 실제 설치를 수행하십시오. 여기에서 언어의 선택, 주변 장치 드라이버 모듈 설정, (CD/DVD 설치 이미지 세트에서 설치하지 않은 경우) 나머지 설치 파일을 데비안 서버에서 직접 검색하도록 네트워크 연결 설정, 하드 디스크 파티션 작업 그리고 베이스 시스템 설치를 합니다. 그리고 태스크를 선택하고 설치합니다. (데비안 시스템의 파티션 설정에 대해 부록 C에서 자세히 설명합니다.)

6. 7장에서, 새로 설치한 베이스 시스템으로 부팅합니다.

시스템 설치가 완료되면 8장 부분을 읽으십시오. 이 장에서는 Unix와 데비안에 대해 더 자세한 정보를 볼 수 있는 위치를 설명하고, 커널을 바꾸는 방법을 설명합니다.

마지막으로, 이 문서에 대한 정보와 이 문서에 참여하는 방법은 부록 E에 있습니다.

1.8 저작권 및 소프트웨어 라이선스 정보

이 문서를 읽고 있는 여러분 대부분은 기업의 상용 소프트웨어에 있는 라이선스를 읽어봤을 것입니다. 보통 그 라이선스에는 구입한 소프트웨어의 복사본 하나를 단일 컴퓨터에서 사용할 수 있다고 쓰여 있습니다. 이 시스템의 라이선스는 그런 라이선스와는 다릅니다. 다니는 학교와 회사의 모든 컴퓨터에 데비안 GNU/리눅스를 설치하셔도 좋고 그러길 바랍니다. 친구에게 빌려주고 컴퓨터에 설치하는 것을 도와 주십시오. 또한 수천 장을 복사하고 판매할 수도 (몇몇 제한이 있을 수 있지만) 있습니다. 왜냐하면 데비안은 자유 소프트웨어이기 때문입니다.

소프트웨어를 자유(free)라고 부르는 것은 소프트웨어의 저작권이 없다는 뜻은 아니고, 자유 소프트웨어를 배포하는 설치 미디어가 무료라는 뜻도 아닙니다. 자유소프트웨어는, 부분적인 의미로는 프로그램의 라이선스에서 프로그램을 배포하고 사용할 권리에 대해 비용을 지불할 필요가 없다는 의미입니다. 또 자유 소프트웨어는 누구라도 소프트웨어를 확장하고 개작하고 수정할 수 있으면서, 그렇게 바꾼 결과물도 배포할 수 있습니다.

> 참고
>
> 
>
> 데비안 프로젝트에서는 사용자의 실용적 필요 때문에 우리의 자유 소프트웨어의 기준에 맞지 않는 패키지도 사용할 수 있습니다. 이 패키지는 공식적인 배포판의 일부가 아니지만, 데비안 미러의 **contrib**나 **non-free** 미러 사이트 또는 서드파티 CD/DVD-ROM에서 구할 수 있습니다. Debian FAQ의 "데비안 FTP 아카이브" 부분을 참조하십시오.

시스템에 들어있는 프로그램의 상당 부분은 "GPL"로 알려진 GNU 일반 공중 안내서(General Public License, GPL)에 따라 이용 허락되고 있습니다. GPL 라이선스의 프로그램의 바이너리를 배포할 때는 반드시 프로그램 소스 코드를 이용 가능하게 해야 합니다. 사용자가 소프트웨어를 수정할 수 있도록 보장하는 것입니다. 이 규정에 따라 그러한 패키지의 모든 소스 코드가[1] 데비안 시스템에 들어 있습니다.

데비안에 수록된 프로그램의 저작권과 소프트웨어 라이선스의 형식은 그 밖에도 여러가지가 있습니다. 각 프로그램의 저작권과 라이선스는 패키지를 설치한 다음 `/usr/share/doc/패키지-이름/copyright` 파일을 보면 찾을 수 있습니다.

라이선스에 대한 더 자세히 알고 싶으시면, 또 어떤 소프트웨어가 main 배포판에 포함될 정도로 충분히 자유 소프트웨어인지 판별하는 데비안의 기준에 대해서 알아 보시려면 데비안 자유 소프트웨어 기준을 참조

[1] 데비안 소스 패키지를 찾고 및 배포 방법과 바이너리를 빌드하는 방법에 대한 자세한 내용은 Debian FAQ의 "Debian 패키지 관리 시스템의 기본"를 참조하십시오.

하십시오.

가장 중요한 법적인 고지는, 이 소프트웨어는 어떠한 보증도 하지 않는다는 것입니다. 이 소프트웨어를 만든 프로그래머는 공동체의 이익을 위해 소프트웨어를 만든 것입니다. 어떤 목적에 대해서도 소프트웨어의 적합성을 보장하지 않습니다. 하지만 소프트웨어가 자유소프트웨어이므로, 여러분에 목적에 맞게 소프트웨어를 수정하는 권리는 여러분에 있습니다. 또한 이런 방법으로 다른 사람이 소프트웨어를 확장하여 생긴 변화로 얻은 이익을 누릴 권리도 여러분에게 있습니다.

제 2 장

시스템 요구 사항

이 절에서는 데비안을 시작하는데 필요한 하드웨어 정보를 다루고 있습니다. 또 GNU 및 Linux에서 지원되는 하드웨어에 관한 더 자세한 정보에 대한 링크도 제공했습니다.

2.1 지원하는 하드웨어

데비안은 Linux 및 kFreeBSD 커널과 GNU 툴셋을 쓰는데 필요한 것 외에는 특별한 하드웨어를 요구하지 않습니다. 그러므로 Linux 및 kFreeBSD 커널, libc, **gcc**등이 포팅되어 있고, 데비안 포팅이 존재하면 아키텍처나 플랫폼에서 데비안을 작동시킬 수 있습니다. 이미 데비안 GNU/리눅스에서 테스트되는 32-bit hard-float ARMv7 아키텍처 시스템의 자세한 내용은 https://www.debian.org/ports/arm/에있는 포팅 페이지를 참조하십시오.

여기서는 32-bit hard-float ARMv7 아키텍처에서 지원하는 여러가지 하드웨어를 모두 설명하지는 않고, 일반적인 정보만 설명한 다음 추가 정보가 들어 있는 웹사이트를 안내해 놓았습니다.

2.1.1 지원하는 아키텍처

데비안 GNU/리눅스 11 릴리스는 9개의 주요 아키텍처와 "기종"이라는 각 아키텍처 변형을 지원합니다.

아키텍처	데비안의 명칭	서브 아키텍처	기종
AMD64 및 인텔 64	amd64		
인텔 x86 기반	i386	일반 x86 컴퓨터	일반
		Xen PV 도메인 전용	xen
ARM	armel	Marvell Kirkwood and Orion	marvell
ARM, 하드웨어 FPU 포함	armhf	멀티플랫폼	armmp
64비트 ARM	arm64		
64비트 MIPS (리틀 엔디안)	mips64el	MIPS Malta	5kc-malta
		Cavium Octeon	octeon
		Loongson 3	loongson-3
32비트 MIPS (리틀 엔디안)	mipsel	MIPS Malta	4kc-malta
		Cavium Octeon	octeon

아키텍처	데비안의 명칭	서브 아키텍처	기종
		Loongson 3	loongson-3
Power Systems	ppc64el	IBM POWER8 또는 그 이후 시스템	
64비트 IBM S/390	s390x	VM-reader 및 DASD에서 IPL	generic

이 문서는 Linux커널을 이용한 32-bit hard-float ARMv7 아키텍처에서의 설치를 다루고 있습니다. 데비안이 지원하는 다른 아키텍처에 관한 정보를 찾고있다면, 데비안 포팅 페이지를 참조하십시오.

2.1.2 3가지 ARM 포트

ARM 아키텍쳐는 발전해 왔고 현대적인 ARM 프로세서는 과거 모델에 없는 기능이 들어 있습니다. 그러므로 데비안에서는 다음 3가지 ARM 포트를 통해 여러가지 종류의 시스템을 최대한 지원합니다:

- 데비안/armel은 오래된 32비트 ARM 프로세서로 하드웨어 부동 소수점 기능(FPU)이 없는 프로세서이고,
- 데비안/armhf는 최소한 ARMv7 아키텍쳐를 구현하고 ARM vector floating point specification의 버전3를(VFPv3) 구현한 최근의 ARM 프로세서에서만 동작합니다. 이 모델의 프로세서에 들어 있는 확장된 기능을 사용하므로 성능이 높습니다.
- 데비안/arm64는 최소한 ARMv8 아키텍쳐를 구현한 64비트 ARM 프로세서입니다.

현재 사용 가능한 ARM CPU는 빅이든 리틀이든 둘 중 하나의 엔디안에서 동작하지만, 실제로 대부분은 리틀 엔디안에서 동작합니다. 데비안/arm64, 데비안/armhf, 데비안/armel도 리틀 엔디안 시스템만 지원합니다.

2.1.3 ARM CPU 설계의 다양함과 복잡한 지원

ARM 시스템은 i386/amd64 기반 PC 아키텍쳐보다 하드웨어가 훨씬 더 다른 점이 많습니다. 그러므로 지원에 복잡한 부분이 많이 있습니다.

ARM 아키텍쳐는 "시스템온칩"(SoC, system on chip)에서 주로 사용합니다. 이 SoC는 여러 회사에서 여러가지 종류의 하드웨어를 집어넣도록 설계하고, 이 중에는 시스템이 부팅할 때 필요한 아주 기본적인 하드웨어도 있습니다. 시스템 펌웨어 인터페이스는 날이 갈수록 표준화가 되었지만, 특히 오래 된 하드웨어의 경우 펌웨어/부팅 인터페이스가 아주 많이 다릅니다. 그래서 이러한 시스템에서는 여러가지 시스템 수준의 저수준 문제를 (PC에서처럼 메인보드의 BIOS/UEFI가 처리하는 게 아니라) 리눅스 커널에서 해결해야 합니다.

리눅스 커널에서 ARM을 지원하기 시작할 때는, 이 하드웨어 다양성 문제 때문에 PC에서처럼 "만능" 커널을 쓸 수 없고 각 ARM 시스템마다 별도의 커널이 필요했습니다. 이런 방식으로는 여러 종류의 시스템에 대응할 수 없기 때문에 여러 ARM 시스템에서 동작할 수 있는 단일 ARM 커널을 사용하는 작업이 되었습니다. 최근의 ARM 시스템에서는 그러한 멀티플랫폼 커널을 사용할 수 있게 지원하지만, 오래된 시스템에서는 아직도 별도의 전용 커널이 필요합니다. 이런 이유로 표준 데비안 배포판은 지정된 몇 개의 오래된 ARM 시스템과 최근 멀티플랫폼("armmp"라고 부르는) 커널이 지원하는 최근 시스템만 데비안/armhf에서 지원합니다.

2.1.4 데비안/armhf가 지원하는 플랫폼

다음 시스템이 데비안/armhf에서 멀티플랫폼(armmp) 커널로 동작한다고 알려져 있습니다:

Freescale MX53 Quick Start Board (MX53 LOCO Board) IMX53QSB는 i.MX53 SoC를 사용하는 개발 보드입니다.

Versatile Express Versatile Express는 ARM에서 나온 개발 보드로, 여러가지 CPU 쪽보드를 사용하는 베이스 보드입니다.

몇몇 Allwinner sunXi 기반 개발 보드 및 임베디드 시스템 armmp 커널은 Allwinner A10(아키텍쳐 코드네임 "sun4i", A10s/A13 (아키텍쳐 코드네임 "sun5i"), A20 (아키텍쳐 코드네임 "sun7i"), A31/A31s (아키텍쳐 코드네임 "sun6i"), A23/A33 ("sun8i" 패밀리의 일부) SoC 기반의 몇몇 개발 보드와 임베디드 시스템을 지원합니다. 완전한 설치 지원은 (설치 프로그램에서 준비된 SD 카드 이미지 포함) 다음 sunXi 기반 시스템에서 가능합니다:

- Cubietech Cubieboard 1 + 2 / Cubietruck
- LeMaker Banana Pi 및 Banana Pro
- LinkSprite pcDuino 및 pcDuino3
- Olimex A10-Olinuxino-LIME / A20-Olinuxino-LIME / A20-Olinuxino-LIME2 / A20-Olinuxino Micro / A20-SOM-EVB
- Xunlong OrangePi Plus

Allwinner sunXi 기반 장치의 시스템 지원은 메인라인 리눅스 커널에 들어 있는 드라이버와 디바이스 트리 정보에 한정됩니다. 벤더 전용 커널 (Allwinner 커널 등) 및 안드로이드 기반의 linux-sunxi.org 3.4 커널은 데비안에서 지원하지 않습니다.

메인라인 리눅스 커널은 Allwinner A10, A10s/A13, A20, A23/A33, A31/A31s SoC에서 일반적으로 시리얼 콘솔, 이더넷, SATA, USB, MMC/SD 카드를 지원합니다. 로컬 디스플레이(HDMI/VGA/LCD) 및 오디오 하드웨어에 대한 지원 수준은 시스템마다 다릅니다. 대부분의 시스템의 경우, 커널에 네이티브 그래픽 드라이버가 없지만 대신에 "simplefb" 인프라스트럭쳐를 사용해 여기서 부트로더가 디스플레이를 초기화하고 커널이 초기화된 프레임버퍼를 재사용합니다. 이 방법은 보통 잘 동작하지만, 일정 한계가 있습니다. 디스플레이 해상도를 중간에 바꾸지 못하고 디스플레이에 대한 전원 관리가 불가능합니다.

보드상의 플래시 메모리는 대용량 저장 장치로 사용되도록 의도한 것이고, sunXi 기반 시스템에서 두 가지 형태가 있습니다. 하나는 일반 NAND 플래시이고, 또 하나는 eMMC 플래시입니다. 예전 sunXi 기반 보드의 보드상 플래시는 일반 NAND 플래시를 사용하고 메인라인 커널에서는 지원하지 않으므로 데비안에서도 지원하지 않습니다. 최근 시스템은 일반 NAND 플래시 대신 eMMC 플래시를 사용합니다. eMMC 플래시 칩은 기본적으로 빠르고 빼는 게 불가능한 SD 카드처럼 보이고 일반 SD 카드와 마찬가지로 지원됩니다.

설치 프로그램에 위 목록에 없는 여러 sunXi 기반 시스템을 기초적으로 지원합니다. 하지만 데비안 프로젝트에서 해당 하드웨어를 사용해 볼 수가 없어서 그러한 시스템은 대부분 테스트되지 않았습니다. 이 시스템에 대해 빌드된 SD 카드 이미지는 제공하지 않습니다. 그렇게 제한되게 지원하는 개발 보드는 다음과 같습니다:

- Olimex A10s-Olinuxino Micro / A13-Olinuxino / A13-Olinuxino Micro

- Sinovoip BPI-M2 (A31s 기반)
- Xunlong Orange Pi (A20 기반) / Orange Pi Mini (A20 기반)

위 목록에 있는 SoC와 시스템에 추가로, Allwinner H3 SoC 및 이에 기반한 여러 보드를 제한적으로 지원합니다. H3에 대한 메인라인 커널 지원은 데비안 9 릴리스 프리즈 당시 아직 진행 중이라서, 설치 프로그램은 H3 기반 시스템에서 시리얼 콘솔, MMC/SD, USB 호스트 컨트롤러만 지원합니다. 보드에 있는 이더넷 포트에 대한 드라이버가 아직 없으므로, 네트워크는 USB 이더넷 어댑터나 USB 와이파이 동글을 이용해야 합니다. 설치 프로그램이 제한적으로 지원하는 그러한 H3 기반 시스템은 다음과 같습니다:

- FriendlyARM NanoPi NEO
- Xunlong Orange Pi Lite / Orange Pi One / Orange Pi PC / Orange Pi PC Plus / Orange Pi Plus / Orange Pi Plus 2E / Orange Pi 2

NVIDIA Jetson TK1 NVIDIA Jetson TK1은 Tegra K1 (Tegra 124라고도 알려짐) 칩 기반으로 한 개발보드입니다. Tegra K1에는 쿼드코어 32비트 ARM Cortex-A15 CPU와 192개의 CUDA 코어가 탑재된 Kepler GPU가 (GK20A) 들어 있습니다. 기타 Tegra 124 기반 시스템도 동작할 수 있습니다.

Seagate Personal Cloud 및 Seagate NAS Seagate Personal Cloud 및 Seagate NAS는 Marvell의 Armada 370 플랫폼에 기반한 NAS 장치입니다. 데비안은 Personal Cloud (SRN21C), Personal Cloud 2-Bay (SRN22C), Seagate NAS 2-Bay (SRPD20), Seagate NAS 4-Bay (SRPD40) 장치를 지원합니다.

SolidRun Cubox-i2eX / Cubox-i4Pro Cubox-i 시리즈는 프리스케일 i.MX6 SoC 패밀리의 작은 큐브 모양의 시스템입니다. Cubox-i 시리즈의 시스템 지원은 메인라인 리눅스 커널에 있는 드라이버와 디바이스트리 정보에 한정되어 있습니다. Cubox-i용 프리스케일 3.0 커널 시리즈는 데비안에서 지원하지 않습니다. 메인라인 리눅스에서 사용할 수 있는 드라이버는 시리얼 콘솔, 이더넷, USB, MMC/SD 카드, HDMI를 통한 기본적인 로컬 디스플레이 (콘솔 및 HDMI) 기능입니다. 추가로 Cubox-i4Pro의 eSATA 포트를 지원합니다.

Wandboard Wandboard Quad, Dual, Solo는 프리스케일 i.MX6 SoC 기반의 개발 보드입니다. 시스템 지원은 메인라인 리눅스 커널에 있는 드라이버와 디바이스트리 정보에 달려 있습니다. wandboard.org 사이트에 있는 Wandboard용 프리스케일 3.0 및 3.10 커널 시리즈는 데비안에서 지원하지 않습니다. 메인라인 리눅스에서 사용할 수 있는 드라이버는 시리얼 콘솔, HDMI를 통한 기본적인 로컬 디스플레이, 이더넷, USB, MMC/SD, SATA (Quad만 지원), 아날로그 오디오 기능입니다. 기타 오디오 옵션 (S/PDIF, HDMI 오디오) 및 내장 무선랜/블루투스 모듈은 테스트되지 않았고 데비안 9에서 지원하지 않습니다.

보통 ARM 멀티플랫폼 기능을 사용하면 위 목록에 없는 armhf 시스템에서 `debian-installer`를 실행할 수 있습니다. 단 `debian-installer`가 사용하는 커널에서 대상 시스템의 구성 요소 및 device-tree 파일을 사용할 수 있어야 합니다. 이 경우 설치 프로그램에서 유저랜드 프로그램까지 동작하도록 설치할 수 있습니다. 하지만 시스템이 부팅 가능하게 만들지는 못합니다. 부팅 가능하게 만드려면 하드웨어 전용 정보가 필요하기 때문입니다.

이러한 시스템에서 `debian-installer`를 사용할 때 설치가 끝나고 시스템이 부팅 가능하도록 수동으로 설정해야 할 수도 있습니다. 예를 들어 `debian-installer`에서 시작한 셸에서 필요한 명령어를 실행하든지 해야 합니다.

2.1.5 다중 프로세서

멀티 프로세서 지원(“대칭 멀티 프로세싱” 또는 SMP라고 부르는)은 이 아키텍처에 사용할 수 있습니다. 데비안 11 표준 커널 이미지는 SMP-alternatives 지원을 사용하여 컴파일되어 있습니다. 여기서는 커널에서 프로세서(또는 프로세서 코어)의 수를 자동으로 검색해 단일 프로세서 시스템에서는 SMP 기능을 사용하지 않게 됩니다.

한 컴퓨터에서 여러 개의 프로세서를 사용하는 일은 원래는 고사양 서버 시스템에서만 일어나는 일이었지만, 최근에는 “멀티코어” 프로세서가 도입되면서 매우 일반적인 일이 되었습니다. 이런 프로세서에는 물리적인 칩 한 개에 두 개 이상의 “코어”라고 부르는 프로세서 유닛이 들어 있습니다.

2.1.6 그래픽 하드웨어 지원

데비안의 그래픽 장치 지원은 내부에 있는 X.Org의 X11 시스템과 커널이 얼마나 지원하느냐에 달려 있습니다. 데스크톱 환경은 X11을 사용하지만 기본적인 프레임버퍼 그래픽은 커널에 들어 있습니다. 3D 하드웨어 가속이나 동영상 가속 같은 기능이 있는 고급 그래픽 기능을 사용할 수 있느냐 여부는, 시스템의 실제 그래픽 하드웨어 및 필요에 따라 추가 “펌웨어” 파일의 설치에 따라(2.2절 참고) 달라집니다.

거의 모든 ARM 시스템에서는 그래픽 하드웨어를 별도 카드로 연결할 필요 없이 내장되어 있습니다. 일부 시스템에서는 그래픽 카드를 연결할 수 있는 확장 슬롯이 있지만 드문 경우입니다. 헤드리스 시스템의 경우 그래픽이 아예 없는 경우도 흔합니다. 기본적인 프레임버퍼 비디오는 그래픽이 있는 모든 장치에서 동작하지만, 3D 그래픽 가속은 변함없이 바이너리 드라이버가 필요합니다. 이러한 상황은 빠르게 개선되고 있지만, bullseye 릴리스 시점에서 nouveau (엔비디아 테그라 K1 SoC) 및 freedreno (퀄컴 스냅드래곤 SoC) 드라이버만 릴리스에 들어 있습니다. 기타 하드웨어는 자유소프트웨어가 아닌 드라이버가 별도로 필요합니다.

지원하는 그래픽 하드웨어와 포인팅 장치에 대한 자세한 내용은 https://wiki.freedesktop.org/xorg/를 참조하십시오. 또 데비안 11는 X.Org 7.7 버전이 들어 있습니다.

2.1.7 네트워크 연결 하드웨어

Linux 커널이 지원하는 네트워크 인터페이스 카드(NIC)는 모두 설치 시스템에서도 지원합니다. 드라이버 모듈은 일반적으로 자동으로 로드됩니다.

대부분의 내장 이더넷 장치를 지원하고 일부 PCI 및 USB 장치에 대해서는 모듈이 들어 있습니다.

2.1.8 주변 장치 및 기타 하드웨어

Linux는 마우스, 프린터, 스캐너, PCMCIA/CardBus/ExpressCard 및 USB 장치와 같은 다양한 하드웨어에 폭넓게 대응하고 있습니다. 하지만 시스템을 설치할 때 이 장치가 필요하지는 않습니다.

2.2 펌웨어가 필요한 장치

디바이스 드라이버가 있냐 없냐와는 별도로, 펌웨어 혹은 마이크로코드라고 하는 걸 읽어들여야 동작하는 하드웨어가 있습니다. 네트워크 인터페이스 카드의 경우(특히 무선 네트워크 장치의 경우) 이런 하드웨어가 많습니다. 또 일부 USB 장치와 하드 디스크 컨트롤러까지도 펌웨어가 필요하기도 합니다.

다수의 그래픽 카드에서는 펌웨어 없이도 기본적인 기능은 동작하지만, 고급 기능을 사용하려면 펌웨어를 시스템에 설치해야 합니다. 어떤 경우에는 성공적으로 설치했다고 해도 설치한 시스템으로 재시작했을 때

검은색 화면이나 알아 보기 힘든 화면만 나타나기도 합니다. 그런 일이 생기면, 그래도 로그인해서 몇 가지 피해갈 방법을 시도해 볼 수 있습니다(6.4.3절 참고).

오래 전 장치에서는 동작하는데 펌웨어가 필요한 경우 제조사가 장치 내부의 EEPROM/플래시 칩 안에 펌웨어를 저장해 놓습니다. 최근의 장치에서는 이제 이런 방식으로 펌웨어를 저장하지 않고, 시스템이 부팅할 때마다 호스트 운영체제에 들어 있는 펌웨어 파일을 해당 장치로 업로드합니다.

대부분 이 펌웨어는 데비안 GNU/리눅스 프로젝트의 기준에 따르면 자유롭지 않은 소프트웨어이기 때문에 메인 배포판이나 설치 시스템에 들어 있지 않습니다. 장치 드라이버가 배포판에 들어 있고 펌웨어를 데비안 GNU/리눅스에서 합법적으로 배포할 수 있는 경우, 아카이브의 non-free 섹션에 별도 패키지로 들어 있기도 합니다.

하지만 그렇다고 해서 이러한 하드웨어를 설치할 때 사용할 수 없는 건 아닙니다. 데비안 GNU/리눅스 5.0부터 `debian-installer`는 USB 메모리같은 이동식 장치에서 펌웨어 파일이나 펌웨어가 들어 있는 패키지를 읽어들일 수 있습니다. 설치할 때 펌웨어 파일이나 패키지를 읽어들이는 방법은 6.4절 부분을 참고하십시오.

`debian-installer`에서 펌웨어 파일 프롬프트를 표시했을 때 이 펌웨어 파일이 없거나 이 자유롭지 않은 펌웨어 파일을 설치하고 싶지 않은 경우, 펌웨어를 읽어들이지 않고도 계속 진행을 시도할 수 있습니다. 드라이버가 추가 펌웨어를 필요한 경우에도, 일부 특정 상황에서만 필요하고(예를 들어 tg3 드라이버는 특정 네트워크 카드 기종에서만 펌웨어가 필요합니다) 펌웨어 없이도 대부분 시스템에서 동작하는 경우도 있습니다.

2.3 GNU/Linux에 적합한 하드웨어 구입

데비안 혹은 다른 GNU/Linux 배포판을 사전 설치 시스템을 출하하고있는 업체도 있습니다. 약간 여분의 돈을 가지고 갈지도 모르지만 어느 정도 안심 할 수 있습니다. 이 하드웨어는 GNU/Linux에서 제대로 지원되고있는 것을 확신하기 때문입니다.

Linux가 번들된 시스템을 구입하는 경우에도, 아니면 중고 시스템을 구입하는 경우에도 그 하드웨어 Linux 커널에서 지원되고 있는지 다시 한 번 확인하는 것이 중요합니다. 위의 참고 자료에 하드웨어가 언급되어 있는지 확인하십시오. (있다면) 구입 영업 사원은 Linux 시스템을 구매하는 것을 전합시다. 또한, Linux에 우호적인 하드웨어 업체를 지원해보세요.

2.3.1 독점적이거나 폐쇄된 하드웨어 피하기

일부 하드웨어 제조 업체는 드라이버를 어떻게 만드는지 알려주지 않습니다. 또 비공개 협약(NDA) 없이는 문서를 보여주지 않아서 드라이버 소스 코드를 공개할 수 없게 만듭니다. 소스 코드 공개는 자유 소프트웨어의 핵심적인 부분입니다. 문서에 접근할 권한이 없으므로, 이런 장치는 Linux에서 동작하지 않습니다.

운영체제와 그 장치 드라이버가 일정한 기능을 하는 장치와 통신하는 방법에 대한 표준이(또는 업계의 사실상 표준이) 있습니다. 그러한 표준 또는 사실상 표준에 따르는 장치는 한 개의 장치 드라이버로 모두 동작하고 특정 장치를 위한 드라이버가 필요하지 않습니다. 일부 하드웨어(예를 들어 키보드나 마우스 같은 USB "휴먼 인터페이스 장치", USB 메모리와 메모리 카드리더 같은 USB 저장 장치)의 경우 이런 원칙이 아주 잘 동작하고 사실상 시장에서 판매되는 모든 장치가 표준에 맞습니다.

그 밖의 경우, 특히 프린터는 불행히도 이런 방식으로 동작하지 않습니다. 여러가지 프린터가(사실상의) 표준 컨트롤 언어로 일부 기능이 동작하긴 하지만, 일부 모델은 비공개 컨트롤 명령이 있어야 동작합니다. 이 비공개 명령은 문서가 없으므로 자유 소프트웨어 운영 체제에서 사용할 수 없거나, 제조사가 제공한 비공개 드라이버를 사용해야 합니다.

그런 하드웨어는 구입할 때는 제조사가 제공한 비공개 드라이버가 있더라도, 드라이버 지원에 따라 하드웨어의 수명이 길지 않습니다. 오늘날 상품의 주기가 짧아졌으므로 소비자용 제품이 단종되고, 제조사의 드라이버 업데이트가 더 이상 없는 경우는 흔히 일어납니다. 시스템 업데이트 이후 과거의 비공개 드라이버가 더이상 동작하지 않는 경우, 정상적인 장치를 드라이버 지원 중단 때문에 사용할 수 없는 상황이 벌어지는 데다가 이 상황에서 할 수 있는 일이 아무 것도 없습니다. 그러므로 사용하는 운영체제가 무엇이든 간에 이러한 폐쇄된 하드웨어는 애초에 구입하지 말아야 합니다.

이런 상황을 개선하려면, 폐쇄된 하드웨어 제조사가 문서 및 관련 자료를 공개하도록 요구하십시오. 그러면 이 하드웨어의 자유 소프트웨어 드라이버를 만들 수 있습니다.

2.4 설치 미디어

여기서는 데비안을 설치할 때 어떤 매체를 사용할 것인지를 결정하는 데 도움이 될 것입니다. 전체를 미디어에 관련해 얘기하는 장(4장)이 있고, 여기서 각 미디어에 대해서 장점과 단점을 설명합니다. 그 부분에서 다시 이 페이지를 참조할 수도 있습니다.

2.4.1 CD-ROM/DVD-ROM/BD-ROM

광학 디스크를 사용한 설치는 대부분 아키텍쳐에서 지원합니다.

2.4.2 네트워크

설치할 때 필요한 파일을 가져오는 데 네트워크를 이용할 수 있습니다. 네트워크 사용 여부는 설치 방법에 따라 달라지고, 설치 중에 질문에 어떻게 답했느냐에 따라 달라집니다. 설치 시스템은 HTTP나 FTP를 사용할 수 있는 거의 모든 종류의 네트워크 연결을 지원합니다. (PPPoE는 지원하지만 ISDN이나 PPP는 지원하지 않습니다.) 설치를 마친 다음에는 ISDN이나 PPP를 사용하도록 설정할 수 있습니다.

CD/DVD나 USB 메모리같은 로컬 저장 장치 없이도 네트워크에서 설치 시스템을 부팅할 수도 있습니다. netboot에 필요한 환경이 구축되어 있다면(즉 네트워크에 DHCP와 TFTP 서비스가 동작하고 있으면), 다수의 컴퓨터에 간단하고 빠르게 운영체제를 설치할 수 있습니다. 필요한 환경을 구축하려면 어느 정도 기술 전문 지식이 필요하므로, 초보자들에게는 권하지 않습니다.

또 다른 방법으로 디스크가 없이(diskless) 설치하고, 네트워크를 사용해 랜과 NFS 마운트에서 부팅할 수도 있습니다.

2.4.3 하드디스크

아키텍처에 따라서는 하드 디스크에서 직접 설치 시스템을 부팅하는 것도 한 가지 방법입니다. 이렇게 하려면 설치 프로그램을 하드 디스크에 복사할 수 있는 다른 운영 체제가 있어야 합니다. 이 방법은 다른 설치 방법이 불가능한 특별한 경우가 아니면 권하지 않습니다.

2.4.4 유닉스 계열 혹은 GNU 시스템

다른 유닉스 계열 시스템이 있다면, (뒤에서 설명하겠지만) 그 시스템을 이용해서 `debian-installer` 없이도 데비안 GNU/리눅스를 설치할 수 있습니다. 이렇게 다른 시스템을 이용하는 방법은 지원하지 않는 하드웨어에 설치하거나 다운타임을 용납할 수 없는 호스트에 유용할 수 있습니다. 이러한 방법에 관심이 있다면,

바로 D.3절 부분으로 넘어가십시오. 이 설치 방식은 다른 설치 방법이 불가능할 때 고급 사용자의 경우에만 사용하길 권합니다.

2.4.5 지원하는 저장 장치

데비안 설치 프로그램에 들어 있는 커널은 최대한 많은 시스템에서 동작할 수 있게 빌드되어 있습니다.

2.5 메모리 및 디스크 공간 요구 사항

일반적인 설치를 하려면 최소한 메모리가 190MB만큼은 있어야 하고 하드 디스크 공간이 920MB만큼 있어야 합니다. 이 숫자는 정말 최소한의 숫자입니다. 실제로 사용할 만한 수준이 어느 정도인지 알고 싶으면, 3.4절 부분을 참고하십시오.

설치 프로그램은 보통 그러한 저용량 메모리 시스템에서 실행할 수 있도록 자동으로 메모리 절약 기법을 사용합니다. 하지만 시험이 덜 된 아키텍쳐에서는 그러한 기법을 제대로 사용하지 않고 놓칠 수도 있습니다. 하지만 **lowmem=1** 또는 **lowmem=2** 부팅 파라미터를 수동으로 사용하면 이 기능을 사용할 수 있습니다. (6.3.1.1절 및 5.3.2절 위치도 참고하십시오.)

메모리나 빈 디스크 공간이 작은 시스템에 설치할 수도 있지만 전문적인 사용자만 하기를 권장합니다.

제 3 장

데비안 GNU/리눅스를 설치하기 전에

이 장에서는 설치 프로그램을 부팅하기 전에, 데비안 설치 준비 사항에 대해 다룹니다. 여기에서는 데이터 백업, 하드웨어에 대한 정보 모으기, 기타 필요한 정보 찾기와 같은 것을 포함합니다.

3.1 설치 과정 개요

먼저 시스템을 다시 설치하는 것에 대해 설명합니다. 데비안에서 시스템 전체를 다시 설치해야 할 상황은 아주 드뭅니다. 다시 설치해야 하는 경우는 하드 디스크의 기계적인 고장이 대부분일 것입니다.

흔히 사용하는 많은 운영 체제에서는 치명적인 문제점이 발생하거나 새로운 운영 체제의 버전으로 업그레이드하려는 경우 완전히 새로 설치해야 합니다. 처음부터 완전히 새로 설치할 필요는 없다고 해도, 프로그램이 새로운 운영 체제에서 제대로 동작하려면 프로그램을 다시 설치해야 합니다.

데비안 GNU/리눅스에서는 잘되지 않는 경우, OS를 대체하지 않고 교체할 수 는 케이스가 훨씬 많습니다. 업그레이드 때 전부 다 설치 필요 없고, 항상 그 자리에서 업그레이드할 수 있습니다. 또한 OS의 릴리스가 계속해서 프로그램을 항상 호환 합니다. 프로그램의 새로운 버전이 최신에 의존하는 소프트웨어를 요구하는 경우 데비안 패키지 시스템은 필요한 소프트웨어를 자동으로 확인하고 확실하게 설치합니다. 다시 설치하지 않도록 힘써 왔으며, 다시 설치하지 않으면 안된다고하는 것은 최후의 수단이라는 것이 포인트입니다. 설치 프로그램은 기존 시스템 위에 다시 설치하도록 설계되지 않았습니다.

다음은 설치 과정에서 해야 할 단계입니다.

1. 설치하려는 하드 디스크에 들어 있는 데이터나 문서를 백업하십시오.
2. 설치를 시작하기 전에 해당 컴퓨터에 대한 정보와 필요한 문서를 모으십시오.
3. 하드 디스크의 데비안 파티션에 사용 가능한 공간을 확보하십시오.
4. 컴퓨터에 필요한 설치 프로그램 소프트웨어와 특별히 필요한 드라이버 파일의 위치를 확인하여 다운로드하십시오.
5. 설치 프로그램이 부팅할 수 있도록 CD/DVD/USB 메모리와 같은 부팅 미디어를 설정하거나, 네트워크 부팅 환경을 만드십시오.
6. 설치 시스템을 부팅하십시오.
7. 설치 언어를 선택하십시오.

8. 이더넷 네트워크 연결이 있으면 활성화하십시오.
9. 데비안을 설치할 파티션을 만들고 마운트합니다.
10. 베이스 시스템의 자동 다운로드/설치/설정을 지켜보십시오.
11. 추가 소프트웨어를 선택하고 설치하십시오.
12. 데비안 GNU/리눅스 및 기존 시스템을 시작할 수 있는 부트로더를 설치하십시오.
13. 새로 설치한 시스템을 처음으로 시작하십시오.

설치할 때 문제가 발생할 때를 대비해서, 각 단계가 어떤 패키지와 상관이 있는지 알아 두는 게 좋습니다. 이 설치 단계의 주요 소프트웨어를 소개하면:

설치 소프트웨어인 `debian-installer`는 이 안내서에서 가장 중점적으로 다룹니다. `debian-installer`는 하드웨어를 찾아서 적당한 드라이버를 읽어들이고, `dhcp-client`를 이용해 네트워크 연결을 설정하고, `debootstrap`을 실행해 베이스 시스템 패키지를 설치하고, `tasksel`로 특정 소프트웨어를 추가로 설치합니다. 이 외에 많은 소프트웨어가 각 단계에서 작은 역할을 담당하고 있지만, 새 시스템이 처음 시작하기까지 전까지는 `debian-installer`가 작업을 마칩니다.

필요에 따라 시스템을 조정하려면, `tasksel`을 이용해 웹서버 또는 데스크톱 환경과 같은 미리 정의한 다양한 번들 소프트웨어를 설치할 수 있도록 선택할 수 있습니다.

설치할 때 중요한 옵션의 하나가 그래픽 데스크톱 환경을 설치할 지 옵션입니다. 그래픽 데스크톱 환경은 X 윈도우 시스템과 그래픽 데스크톱 환경 하나로 구성되어 있습니다. "데스크톱 환경" 태스크를 선택하지 않으면, 아주 기본적인 명령행 기반 시스템만 설치합니다. 데스크톱 환경 태스크 설치는 옵션입니다. 데스크톱 환경 태스크를 설치하면 텍스트 전용 시스템과 비교해 큰 디스크 공간을 차지하기 때문이고, 또 많은 데비안 GNU/리눅스 시스템은 서버로 동작하기 때문에 동작하는 데 그래픽 사용자 인터페이스가 전혀 필요없기 때문입니다.

X 윈도우 시스템은 `debian-installer`와는 완전히 별도의 소프트웨어이고, 실제로 훨씬 더 복잡합니다. X 윈도우 시스템의 문제점 해결은 이 안내서가 다루는 범위를 벗어납니다.

3.2 기존 데이터를 백업하십시오!

시작하기 전에 지금 시스템에 있는 모든 파일을 백업해 두십시오. 원래 설치된 운영 체제가 아닌 운영 체제를 처음으로 설치하는 거라면, 데비안 GNU/리눅스의 루트로 쓸 디스크를 다시 파티션해야 합니다. 파티션 프로그램으로 어떤 프로그램을 사용하든 간에, 디스크를 파티션하면 그 디스크에 있는 모든 파일을 잃어버리는 걸 감수해야 합니다. 데비안 GNU/리눅스 설치에 사용하는 프로그램은 상당히 안정적이고 수년동안 사용해 온 프로그램이지만, 그만큼 강력하기도 해서 조금만 잘못하면 막심한 손해가 발생합니다. 백업을 한 뒤에도 대답을 할 때나 어떤 작업을 할 때 신중을 기하십시오. 일이분만 더 생각하면 수 시간의 불필요한 작업을 예방할 수 있습니다.

멀티 부팅 시스템을 만든다면, 기존 운영 체제의 배포 미디어를 가지고 있으십시오. 보통 그럴 필요가 없겠지만 시스템이 부팅하려면 부트로더를 다시 설치해야 할 수도 있고, 최악의 상황에서는 전체 운영 체제를 다시 설치하고 백업한 내용을 복구해야 할 수도 있습니다.

3.3 필요한 정보

3.3.1 문서

3.3.1.1 설치 안내서

지금 읽고 있는 이 문서는 데비안의 bullseye 릴리스 공식 설치 가이드의 개발 버전입니다. 이것은 다양한 형식과 언어 를 사용할 수 있습니다.

3.3.1.2 하드웨어 문서

하드웨어를 설정하고 이용하는 방법에 대한 유용한 정보가 들어있습니다.

3.3.2 하드웨어 정보가 있는 곳 찾기

보통은 설치 프로그램에서 자동으로 하드웨어를 찾아 냅니다. 하지만 철저히 준비하려면, 설치하기 전에 하드웨어에 대해 잘 알아 두는 게 좋습니다.

하드웨어 정보는 다음에서 알아낼 수 있습니다:

- 하드웨어에 같이 들어 있는 설명서.
- 컴퓨터의 BIOS/UEFI 설정 화면. 컴퓨터가 시작할 때 어떤 키 조합을 누르면 BIOS 설정 화면을 볼 수 있습니다. 보통 **Delete** 또는 **F2** 키이지만, 제조사에 따라 다른 키나 키조합을 사용할 수도 있습니다. 보통 컴퓨터 부팅할 때 어떤 키를 눌러야 설정 메뉴로 들어가는지 메시지를 표시합니다.
- 하드웨어의 케이스 및 포장.
- 파일 관리자에서 표시하는 것과 같은, 다른 운영 체제의 시스템 명령어 및 시스템 도구. 특히 RAM과 하드 드라이브에 대한 정보를 알아내는 데 유용합니다.
- 시스템 관리자 혹은 인터넷 서비스 회사. 여기서는 네트워크 및 전자메일 설정에 관련된 사항을 알 수 있습니다.

표 3.1 설치에 도움이 되는 하드웨어 정보

하드웨어	**필요할 수 있는 정보**
하드 드라이브	용량이 얼마나 되는 지.
	시스템에 붙어 있는 순서.
	IDE (PATA), SATA, SCSI 중 한 가지.
	사용 가능한 빈 공간.
	파티션.
	다른 운영 체제를 설치한 파티션.
네트워크 인터페이스	네트워크 인터페이스의 종류/모델.
프린터	모델 및 제조사.
비디오 카드	타입/모델 및 제조사.

3.3.3 하드웨어 호환성

여러가지 제품이 문제없이 Linux에서 작동합니다. 또한 Linux에서 지원하는 하드웨어는 날이 갈수록 발전하고 있습니다. 하지만 아직도 Linux에서는 다른 OS 만큼 다양한 하드웨어를 지원하지는 않습니다.

Linux의 대부분 드라이버는 특정 제조사의 특정 제품이나 특정 브랜드를 위해 작성된 드라이버가 아니고, 특정 하드웨어/칩셋에 맞춰 작성되었습니다. 한 개의 하드웨어 설계에 맞춰 여러 개의 제품과 브랜드가 있습니다. 보통 칩 제조사는 "레퍼런스 설계"라는 것을 제공하고, 거기에 맞춰 여러가지 제조사가 여러가지 제품과 브랜드를 만들 수 있습니다.

이러한 방식은 장점과 단점이 있습니다. 장점은 한 개의 칩셋 드라이버가, 여러가지 제조사와 여러가지 제품에서 그 칩셋을 계속 사용하는 한 동작한다는 점입니다. 단점은 어떤 제품에 어떤 칩셋이 사용되었는지 정확히 아는 게 쉬운 일만은 아니라는 점입니다. 심지어 장치 제조사는 제품 이름이나 제품 버전을 바꾸지도 않고 하드웨어 구조를 바꿔 버리기도 합니다. 그러므로 시간이 지난 다음에 같은 브랜드나 같은 이름의 제품을 구입하더라도 다른 칩셋을 사용할 수도 있어서 다른 드라이버를 사용해야 하거나, 한 가지 제품만 드라이버가 없을 수도 있습니다.

USB 및 PCI/PCI-Express/ExpressCard 장치의 경우, 어떤 칩셋을 사용하고 있는지 알아내는 방법은 장치 아이디를 살펴보는 방법입니다. 모든 USB/PCI/PCI-Express/ExpressCard 장치는 "공급사"와 "제품"마다 아이디가 있고, 이 두 아이디의 조합이 동일하면 같은 칩셋을 사용하는 같은 제품입니다.

리눅스 시스템에서는 이 아이디는 USB 장치의 경우 **lsusb** 명령, PCI/PCI-Express/ExpressCard 장치는 **lspci -nn** 명령으로 알아볼 수 있습니다. 공급사와 제품 아이디는 콜론으로 구분한 두 개의 16진수 숫자 형태로(예를 들어 "1d6b:0001") 주어집니다.

예를 들어 **lsusb** 출력은: "Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub" 여기서 1d6b는 공급사 아이디이고 0002는 제품 아이디입니다.

이더넷 카드의 **lspci -nn** 출력은: "03:00.0 Ethernet controller [0200]: Realtek Semiconductor Co., Ltd. RTL8111/8168B PCI Express Gigabit Ethernet controller [10ec:8168] (rev 06)". 오른쪽의 각괄호 안에 아이디가 있습니다. 즉 10ec가 공급사 아이디이고, 8168이 제품 아이디입니다.

다른 예로 그래픽 카드 출력이 다음과 같을 수도 있습니다: "04:00.0 VGA compatible controller [0300]: Advanced Micro Devices [AMD] nee ATI RV710 [Radeon HD 4350] [1002:954f]".

윈도우 시스템에서는 윈도우 장치 관리자의 "자세히" 탭에서 볼 수 있습니다. 여기에서 공급사 아이디는 앞에 "VEN_"이 붙고, 제품 아이디에는 앞에 "DEV_"이 붙습니다. 윈도우 7 시스템에서는 장치 관리자의 탭에서 기본값으로는 아이디가 표시되지 않고, "하드웨어 ID" 속성을 선택하면 아이디를 볼 수 있습니다.

인터넷에서 공급사/제품 아이디를 검색할 때, 검색어로 "Linux" 및 "driver"로 검색하면 드라이버의 특정 칩셋 지원에 관한 정보가 잘 나옵니다. 공급사/제품 아이디로 검색해도 쓸만한 결과가 나오지 않으면, 칩 코드로 검색해 보십시오. 칩 코드는 lsusb 및 lspci로 알 수 있습니다. (예를 들어 네트워크 카드는 "RTL8111"/"RTL8168B"가 있고 그래픽 카드는 "RV710"이 있습니다.)

3.3.3.1 라이브 시스템에서 하드웨어 호환성 검사하기

데비안 GNU/리눅스는 일부 아키텍처에서는 "라이브 시스템"라고 말하는 방식으로도 사용할 수 있습니다. 라이브 시스템은 CD나 DVD같은 읽기 전용 미디어에서 부팅해 바로 사용할 수 있는 시스템입니다. 이 방식에서는 컴퓨터에 아무 것도 변경하지 않습니다. 라이브 시스템에서 사용자 설정을 바꿀 수도 있고 프로그램을 추가로 설치할 수도 있지만, 컴퓨터의 램 안에서만 벌어지는 일입니다. 즉 컴퓨터를 껐다가 다시 라이브 시스템으로 부팅하면 모조리 기본값으로 초기화됩니다. 데비안 GNU/리눅스에서 컴퓨터의 하드웨어를 지원하는지 알아볼 때 가장 쉬운 방법은 데비안 라이브 시스템을 돌려보고 사용해 보는 방법입니다.

라이브 시스템은 사용하는데 몇 가지 제약이 있습니다. 첫째는 라이브 시스템에서 안에서 한 작업이 컴퓨터의 램에 들어가기 때문에 램이 충분히 큰 시스템에서만 동작합니다. 그러므로 용량이 큰 소프트웨어를 설치할 경우 메모리 부족으로 실패할 수 있습니다. 하드웨어 호환성 테스트와 관련된 또 제약은 공식 데비안 GNU/리눅스 라이브 시스템에는 자유 소프트웨어로만 구성되어 있다는 점입니다. 즉 자유롭지 못한 펌웨어 파일은 들어 있지 않습니다. 그러한 펌웨어 패키지를 수동으로 설치할 수도 있지만, `debian-installer`와 같은 펌웨어 파일 자동 검색 기능은 없기 때문에 필요할 때마다 수동으로 설치해야 합니다.

사용할 수 있는 데비안 라이브 이미지의 종류는 데비안 라이브 이미지 웹사이트에 있습니다.

3.3.4 네트워크 설정

컴퓨터가 다른 누군가가 관리하고 있는 고정된 네트워크에 연결되어 있다면(이더넷이나 비슷한 연결의 경우, 전화접속이나 PPP는 아님), 네트워크 시스템 관리자에게 이러한 정보를 알아봐야 합니다:

- 호스트 이름(직접 정할 수도 있습니다).
- 도메인 이름.
- 컴퓨터의 IP 주소.
- 네트워크에 사용할 네트마스크.
- 라우팅하는 데 쓸 기본 게이트웨이 시스템의 IP 주소(게이트웨이가 있는 경우).
- 네트워크에서 DNS (Domain Name Service) 서버로 사용할 시스템.

연결된 네트워크가 네트워크 설정에 DHCP(Dynamic Host Configuration Protocol)를 사용한다면 이 정보가 필요 없습니다. 설치할 때 DHCP 서버에서 컴퓨터로 이 정보를 알려줍니다.

DSL이나 케이블 모뎀을(예를 들어 케이블 TV 네트워크를 통해) 통해 인터넷을 사용하고 라우터가(보통 전화 또는 케이블 사업자가 설정한 상태로 설치됩니다) 네트워크 연결을 처리하는 경우, 보통 DHCP를 사용할 수 있습니다.

무선랜/와이파이 네트워크를 사용한다면, 다음을 알아봐야 합니다:

- 무선 네트워크의 ESSID(“네트워크 이름”).
- 이 네트워크에 연결할 때 사용할 WEP 또는 WPA/WPA2 보안 키(보안 키를 사용하는 경우).

3.4 최소 하드웨어 요구 사항 맞추기

컴퓨터의 하드웨어에 대한 정보를 모았으면, 설치하려는 방법대로 그 하드웨어에서 설치할 수 있는 지 확인하십시오.

필요에 따라서는, 아래의 표에 나온 하드웨어보다 성능이 떨어지는 하드웨어로도 설치할 수 있을 수도 있습니다. 하지만 아래 제안을 무시할 경우 대부분 사용자는 짜증나는 문제를 겪게 됩니다.

표 3.2 추천하는 최소 시스템 요구사항

설치 종류	**램(최소)**	**램(추천)**	**하드 드라이브**
데스크톱 없음	256메가바이트	512메가바이트	2기가바이트
데스크톱 포함	1기가바이트	2기가바이트	10기가바이트

최소값은 스왑 기능을 사용하고 liveCD가 아닌 이미지를 사용한다는 가정 하에 최소값입니다. "데스크톱 없음" 값은 그래픽이 아닌 (텍스트 기반) 설치 프로그램을 사용한다는 가정입니다.

실제로 필요한 최소 메모리는 이 표에 나와 있는 숫자보다 훨씬 작습니다. 스왑을 사용하면, 140MB 만큼 작은 메모리에서도 데비안을 설치할 수 있습니다. 필요한 디스크 공간도 마찬가지입니다. 특히 설치할 프로그램을 직접 선택하면 공간을 절약할 수 있습니다. 필요한 디스크 공간에 대한 추가 정보는 D.2절를 참조하십시오.

성능이 낮은 오래된 시스템에도 그래픽 데스크톱 환경을 설치할 수 있지만, 그놈이나 KDE 플라즈마 데스크톱 환경을 설치하지 말고 리소스를 적게 차지하는 윈도우 매니저를 설치하길 권장합니다. 예를 들어 `xfce4`, `icewm`, `wmaker` 등 기타 여러가지가 있습니다.

서버로 설치할 경우 필요한 메모리나 디스크 공간을 예측하기는 사실상 불가능합니다. 그 서버가 어떤 용도로 사용하느냐에 따라 많이 달라지기 때문입니다.

여기 나와 있는 크기는 사용자 파일, 메일, 데이터 등과 같은 데이터는 포함하지 않은 숫자입니다. 사용자의 파일 및 데이터가 차지하는 공간을 잡을 때는 넉넉하게 잡는 게 좋습니다.

데비안 GNU/리눅스 시스템을 원활히 동작시키는 데 필요한 디스크 공간은 권장 시스템 요구 사항에 반영되어 있습니다. 특히, `/var` 파티션에 로그 파일 같은 일반적인 내용뿐만 아니라 데비안 특유의 상태 정보가 들어 있습니다. **dpkg** 파일은(설치 패키지에 대한 정보) 쉽게 40MB를 차지합니다. 또한 **apt**는 설치하기 전에 다운로드한 패키지를 여기에 놓습니다. `/var`에 최소 200MB를 할당해야만 합니다. 그리고 그래픽 데스크톱 환경을 설치하는 경우에는 더 할당 할 것입니다.

3.5 멀티 부팅 시스템에서 미리 파티션하기

디스크 파티션은 디스크를 여러 개의 조각으로 나누는 작업을 말합니다. 각각의 조각은 다른 조각과 독립적입니다. 비유 하자면 집에 벽을 놓는 것과 비슷합니다. 어떤 방에 가구를 놓는다고 해서 다른 방에 영향을 끼치지 않습니다.

시스템에 이미 다른 운영 체제가 전체 디스크를 차지하고 있으면서, 같은 디스크에 데비안도 설치하려는 경우에는 디스크 파티션 분할을 다시 시작해야합니다. 데비안은 윈도우나 맥오에스 파티션에 설치할 수 없습니다. 다른 유닉스 시스템과는 일부 파티션을 공유할 수 있을지도 모르지만, 여기에서는 다루지 않습니다. 적어도 데비안의 root에 사용할 전용 파티션이 필요합니다.

현재 파티션 상태 정보는 현재 운영 체제의 파티션 프로그램을 이용해 알 수 있습니다. 같은 프로그램이 있습니다. 파티션하는 프로그램은 파티션을 바꾸지 않고도 현재 파티션을 표시하는 기능이 있습니다.

이미 파일 시스템이 들어 있는 파티션을 바꾸면 보통 거기에 들어 있는 정보가 모두 망가집니다. 그러므로 파티션 작업을 하기 하기 전에 항상 백업을 만들어야 합니다. 집의 경우에 비유하자면, 벽을 옮기기 전에 모든 가구를 다른 곳에 옮겨 놓아야 가구가 망가지지 않을 겁니다.

최근의 몇몇 운영 체제에는 파티션 내용을 망가뜨리지 않고도 파티션의 위치를 옮기거나 크기를 바꾸는 기능이 있습니다. 이 기능을 이용해 데이터를 잃지 않고도 파티션을 만드는데 필요한 공간을 만들 수 있습니다. 대부분의 경우 이 기능은 잘 동작하지만 디스크 파티션을 변경하는 작업은 본래 위험한 작업이므로 데이터를 모두 백업한 다음 진행해야 합니다.

`debian-installer`에서 파티션을 만들고 삭제하는 일은 `debian-installer`에서도 할 수 있고 기존 운영 체제에서도 할 수 있습니다. 한 가지 지켜야 할 규칙은, 해당 파티션을 사용하는 시스템에서 파티션을 만들도록 하십시오. 예를 들어 데비안 GNU/리눅스에서 사용할 파티션은 `debian-installer` 안에서 만들고, 다른 운영 체제에서 사용할 파티션은 그 운영 체제 안에서 만듭니다. `debian-installer`에서는 Linux이 아닌 파티션도 만들 수 있고, 이렇게 만든 파티션도 다른 운영 체제에서 잘 동작합니다. 하지만 드물게 이것 때문에 문제가

발생하는 경우가 있습니다. 그러므로 확실히 하려면 다른 운영 체제가 사용하는 파티션은 그 운영 체제의 자체 파티션 프로그램을 사용해 만드십시오.

같은 컴퓨터에 여러 개의 OS를 설치하려는 경우, 데비안을 설치하기 전에 다른 OS를 먼저 설치하고 둡시다. Windows 등의 다른 OS를 설치하면 데비안을 시작하는 기능이 파괴되어 버리거나, 혹은 그 OS의 파티션이 아닌 파티션을 다시 포맷하라고 할 수도 있습니다.

이렇게 되더라도 복구할 수도 있고, 이렇게 되지 않도록 피할 수도 있습니다. 하지만 원래 운영 체제를 먼저 설치하면 이런 문제가 없습니다.

3.6 설치하기 전에 할 하드웨어 및 운영 체제 설정

이 절에서는 데비안 설치에 앞서 필요한 하드웨어 설정에 대해 알아보겠습니다. 일반적으로 이 작업은 BIOS/UEFI/시스템 펌웨어 설정을 확인하고 필요하다면 설정을 변경하는 작업입니다. “BIOS/UEFI” 또는 “시스템 펌웨어”는 하드웨어가 사용하는 핵심 소프트웨어로, 전원을 켠 다음에 부팅 과정 동안에 시작되는 가장 중요한 소프트웨어입니다.

3.6.1 ARM 펌웨어

앞에서도 말했듯이, ARM 시스템에서는 아쉽게도 시스템 펌웨어의 표준이 없습니다. 같은 펌웨어를 사용하는 다른 시스템도 동작이 아주 다릅니다. 이는 ARM 아키텍쳐를 사용하는 하드웨어의 상당 부분이 임베디드 시스템이기 때문입니다. 임베디드 시스템에서는 제조사가 특별히 수정된 버전의 펌웨어를 빌드하고 이 펌웨어에는 하드웨어 전용 패치가 들어 있습니다. 아쉽게도 이 제조사는 수정 사항이나 확장한 기능을 상위의 펌웨어 개발자에게 전달하지 않습니다. 그러므로 수정 사항이 새 버전의 원래 펌웨어에서는 빠지게 됩니다.

결과적으로 새로 판매되는 시스템에서도 수년이 지나도 제조사 버전의 수정된 펌웨어를 사용하게 되는데, 그 동안 상위 메인라인의 코드는 엄청나게 발전해서 추가 기능을 제공할 수도 있고 동작이 달라질 수도 있습니다. 또 같은 펌웨어의 제조사 수정 버전이 달라져도 하드웨어의 장치 이름이 일관적이지 않습니다. 그래서 ARM을 사용하는 시스템에서는 제품과 무관한 설치 안내를 하기가 불가능합니다.

3.6.2 데비안 공급 U-Boot (시스템 펌웨어) 이미지

데비안에서 여러가지 armhf 시스템에서 SD 카드에서 U-Boot를 읽어들일 수 있는 U-Boot 이미지를 .../-images/u-boot/ 위치에 제공합니다. U-Boot 빌드는 2가지 형식으로 제공됩니다: 하나는 가공되지 않은 형식의 U-Boot 구성 요소는 또 하나는 SD 카드에 간단히 쓸 수 있는 카드 이미지 파일입니다. 가공되지 않은 U-Boot 구성요소는 고급 사용자용입니다. 추천하는 방법은 SD 카드 이미지 사용입니다. SD 카드 이미지 파일은 <시스템-종류>.sdcard.img.gz 파일로 SD 카드에 다음과 같은 명령으로 쓸 수 있습니다:

```
zcat <시스템-종류>.sdcard.img.gz > /dev/SD_카드_장치
```

어떤 이미지이든, 이미지를 SD 카드에 쓰면 카드의 이전 내용을 모두 엎어쓰니 주의하십시오!

데비안에서 시스템에 맞는 U-Boot 이미지를 제공하는 경우, 판매 회사가 제공하는 U-Boot 대신 그 이미지를 사용하기를 추천합니다. 데비안에 들어 있는 버전이 보통 더 최신으로 더 많은 기능이 들어 있습니다.

3.6.3 U-Boot에서 이더넷 MAC 주소 설정하기

모든 이더넷 인터페이스의 MAC 주소는 보통은 전세계에서 유일해야 합니다. 그리고 기술적으로도 이더넷 브로드캐스트 범위 안에서는 주소가 유일해야 합니다. 이렇게 유일성을 보장하기 위해, 중앙에서 관리하는

모음에서 제조사마다 일정한 블럭의 MAC 주소 묶음을 배정받습니다. (그리고 일정한 사용료를 냅니다.) 그 다음에 판매하는 제품마다 주소를 하나씩 부여합니다.

개발 보드의 경우에는, 제조사에서 이 사용료를 내지 않으려 하기도 해서 유일한 주소가 없을 수도 있습니다. 이 경우 사용자가 직접 MAC 주소를 설정해야 합니다. 이더넷 인터페이스에 MAC 주소가 지정되어 있지 않았을 때, 일부 드라이버는 부팅할 때마다 임의의 MAC 주소를 생성하기도 합니다. 이렇게 동작하는 경우 수동으로 주소를 지정하지 않아도 네트워크를 사용할 수 있습니다. 하지만 예를 들어 DHCP 서버가 클라이언트 MAC 주소에 따라 일정한 IP 주소를 할당하는 경우에는 제대로 동작하지 않습니다.

공식적으로 배정된 MAC 주소와 충돌을 피하려면 "사설(locally administered)" 주소로 예약된 주소가 있습니다. 이 주소는 주소의 첫 바이트에서 두 비트로 정의됩니다. (영문 위키백과의 "MAC address" 글에 잘 설며오디어 있습니다.) 예를 들어, 16진수 "ca"로 시작하는 모든 주소는(예를 들어 ca:ff:ee:12:34:56) 사설 주소입니다.

U-Boot를 시스템 펌웨어로 사용하는 시스템에서는, 이더넷 MAC 주소를 "ethaddr" 환경 변수에 저장합니다. 이 값은 U-Boot 명령 프롬프트에서 "printenv ethaddr" 명령으로 확인해 볼 수 있고, "setenv ethaddr ca:ff:ee:12:34:56" 명령과 같이 설정할 수 있습니다. 값을 설정하면 "saveenv" 명령으로 값을 저장합니다.

3.6.4 U-Boot의 커널/최초 램디스크/디바이스 트리 재배치 문제

예전 버전의 U-Boot를 사용하는 일부 시스템에서는, 부팅 과정에서 메모리에 있는 리눅스 커널, 최초 램디스크, 디바이스 트리를 재배치할 때 문제가 있을 수도 있습니다. 이 경우 U-Boot에서 "Starting kernel ..." 메시지를 표시히지만, 그 뒤에 출력이 없이 시스템이 멈춰 있습니다. 이 문제는 U-Boot v2014.07 버전 이후로 해결되었습니다.

만약 시스템에서 v2014.07보다 오래 된 U-Boot 버전을 사용하고 나중에 그보다 나중 버전으로 업그레이드했다면, U-Boot 업그레이드 후에도 문제가 계속 발생할 수 있습니다. U-Boot를 업그레이드해도 현재 U-Boot 환경 변수를 수정하지는 않고, 추가 환경 변수를(bootm_size) 설정해야 합니다. U-Boot는 환경 변수 데이터가 없이 새로 설치할 경우에만 자동으로 이 환경 변수를 설정합니다. U-Boot 프롬프트에서 "env default bootm_size; saveenv" 명령으로 bootm_size 환경 변수를 U-Boot의 기본값으로 수동 설정할 수도 있습니다.

재배치와 관련된 문제를 피해가는 한 가지 가능한 방법은 "setenv fdt_high ffffffff; setenv initrd_high 0xffffffff; saveenv" 명령을 U-Boot 프롬프트에서 실행해서 최초 램디스크와 디바이스 트리 파일의 재배치를 완전히 막는 방법입니다.

제 4 장

시스템 설치 미디어 구하기

4.1 공식 데비안 GNU/리눅스 설치 이미지

현재 데비안 GNU/리눅스를 설치하는 가장 쉬운 방법은 공식 데비안 설치 이미지 세트로 설치하는 것입니다. 공급 업체에서 CD/DVD 세트를 구입할 수 있습니다(CD 벤더 페이지 참조). 고속 네트워크 연결과 CD/DVD 라이터가 있으면, 데비안 미러 사이트에서 설치 이미지를 다운로드해도 괜찮습니다(자세한 설명은 데비안 CD 페이지 및 데비안 CD FAQ 참고). 그러한 광학 설치 미디어가 있고, 컴퓨터에서 설치 이미지로 부팅할 수 있으면, 5장 부분으로 넘어갈 수 있습니다. 많이 사용되는 파일이 첫번째 CD 또는 DVD 이미지에 들어가도록 많은 노력을 하고 있습니다. 그러므로 기본적인 데스크톱 설치는 첫번째 DVD 또는 첫번째 CD로도 (제한적이긴 하지만) 가능합니다.

CD는 요즘 추세에서는 용량이 그리 크기 않기 때문에, 그래픽 데스크톱 환경 중에서는 첫 번째 CD로 설치할 수 없는 경우도 있습니다. 일부 데스크톱 환경은 설치 중에 나머지 파일을 다운로드할 수 있도록 네트워크 연결이 필요하거나 추가 CD가 필요합니다.

이 점을 기억해 두십시오. 사용하는 설치 미디어에 필요한 패키지가 없는 경우에도, 나중에 (설치를 마친 뒤에) 언제든 데비안 시스템에서 해당 패키지를 설치할 수 있습니다. 어떤 설치 이미지에서 특정 패키지를 찾을 수 있는지는 https://cdimage-search.debian.org/ 페이지를 참고하십시오.

여러분의 컴퓨터가 광학 미디어 부팅을 지원하지 않지만 CD/DVD 세트를 가지고 있다면, 다른 방법으로 네트워크 부팅 방법을 사용할 수 있고, 아니면 커널을 디스크에서 수동으로 읽어들여 설치 프로그램을 부팅할 수 있습니다. 이렇게 부팅할 때 필요한 파일도 디스크에 들어 있습니다. 데비안 네트워크 아카이브와 CD의 폴더 구조는 동일합니다. 그러므로 부팅하는 데 필요한 어떤 파일이 아카이브에서 있다면, 설치 미디어의 같은 하위 디렉터리에서 파일을 찾을 수 있습니다.

설치 프로그램을 부팅하기만 하면, 필요한 다른 파일은 모두 디스크에서 이용할 수 있습니다.

설치 미디어 세트가 없으면, 설치 프로그램의 시스템 파일을 다운로드해서 네트워크로 연결된 컴퓨터에 저장해 놓으십시오. 그러면 이 파일을 이용해 설치 프로그램을 부팅할 수 있습니다.

4.2 데비안 미러에서 파일 다운로드

가장 가까이있는(그래서 가장 빠를 것 같은) 미러 사이트를 찾으려면 데비안 미러 사이트 목록를 참조하십시오.

4.2.1 설치 파일을 찾을 위치

설치 파일은 데비안 미러의 debian/dists/bullseye/main/installer-armhf/current/images/에 있습니다. 각 이미지 이름과 그 용도가 MANIFEST에 설명되어 있습니다.

4.2.1.1 armhf 멀티플랫폼 설치 파일

armhf 멀티플랫폼 커널이(2.1.4절 참고) 지원하는 시스템의 설치 파일은 표준 리눅스 커널 이미지, 표준 리눅스 최초 램디스크 이미지, 시스템 전용 디바이스 트리 파일입니다. 커널 및 TFTP 부팅에 필요한 최초 램디스크 이미지는 .../images/netboot/ 에 있고, 디바이스트리 파일은 .../images/device-tree/ 에 있습니다. 부팅 가능 USB 메모리를 만드는 tar 압축은 .../images/hd-media/ 위치에 있습니다.

여러가지 armhf 플랫폼의 U-Boot 이미지가 .../images/u-boot/ 위치에 있습니다.

4.3 TFTP 네트워크 부팅에 필요한 파일 준비하기

LAN에 연결되어 있다면, 네트워크를 통해 TFTP를 사용해서 다른 컴퓨터에서 부팅할 수도 있습니다. 다른 컴퓨터에서 설치 시스템을 부팅하려고 한다면, 부팅 파일을 특정 위치에 복사해 놓고, 해당 컴퓨터의 부팅을 지원하도록 설정해 놓아야 합니다.

TFTP 서버를 설정해야 합니다. 그리고 많은 컴퓨터에서 DHCP 서버 아니면 RARP 서버, 아니면 BOOTP 서버를 설정해야 합니다.

Reverse Address Resolution Protocol(RARP)은 어떤 클라이언트에게 어떤 IP 주소를 사용해야 하는 지 알려주는 한 방법입니다. 또 다른 방법은 BOOP 프로토콜을 사용하는 것입니다. BOOTP는 컴퓨터에게 그 IP 주소 및 네트워크의 어디에서 부팅 이미지를 가져와야 하는 지 알려주는 IP 프로토콜입니다. DHCP(Dynamic Host Configuration Protocol)는 더 유연하며, BOOTP와 호환되는 확장 기능입니다. 일부 시스템은 DHCP를 이용해야만 설정할 수 있습니다.

Trivial File Transfer Protocol은(TFTP) 부팅 이미지를 클라이언트에게 넘겨줄 때 사용합니다. 이론상 이 프로토콜을 사용한다면 어떤 플랫폼의 어떤 서버라도 사용할 수 있습니다. 여기서는 SunOS 4.x, SunOS 5.x(Solaris), 그리고 GNU/리눅스에서 사용하는 명령을 예로 듭니다.

4.3.1 RARP 서버 준비하기

RARP를 설정하려면, 클라이언트 컴퓨터에 설치되어있는 이더넷 주소(MAC 주소)를 알아야합니다. 이 정보를 모르면 “Rescue”모드를 시작하고 **ip addr show dev eth0** 명령을 사용하십시오.

리눅스 커널을 사용하는 RARP 서버 시스템이나 Solaris/SunOS에서는 **rarpd** 프로그램을 사용합니다. 클라이언트에 대한 이더넷 하드웨어 주소를 “ethers” 데이터베이스에 넣어야 합니다(/etc/ethers 파일이나 NIS/NIS+를 이용). 그 다음에 RARP 데몬을 시작합니다. 다음 명령을(root로) 실행합니다: 대부분 리눅스 시스템이나 SunOS5(Solaris 2)에서는 **/usr/sbin/rarpd -a**, 기타 리눅스 시스템에서는 **/usr/sbin/in.rarpd -a**, SunOS4(Solaris 1)에서는 **/usr/etc/rarpd -a**.

4.3.2 DHCP 서버 준비하기

자유 소프트웨어 DHCP 서버의 하나로 ISC **dhcpd**가 있습니다. 데비안 GNU/리눅스에서는 isc-dhcp-server 패키지를 권장합니다. 다음은 간단한 설정 파일 예제입니다 (/etc/dhcp/dhcpd.conf 파일을 보십시오):

```
option domain-name ”example.com”;
option domain-name-servers ns1.example.com;
option subnet-mask 255.255.255.0;
default-lease-time 600;
max-lease-time 7200;
server-name ”servername”;

subnet 192.168.1.0 netmask 255.255.255.0 {
  range 192.168.1.200 192.168.1.253;
  option routers 192.168.1.1;
}

host clientname {
  filename ”/tftpboot.img”;
  server-name ”servername”;
  next-server servername;
  hardware ethernet 01:23:45:67:89:AB;
  fixed-address 192.168.1.90;
}
```

이 예제에서는, DHCP 서버, TFTP 서버, 네트워크 게이트웨이 역할을 모두 하는 servername이라는 서버가 있다고 가정합니다. domain-name 옵션도 바꾸고, 서버 이름과 클라이언트 하드웨어 주소도 바꿔야 합니다. filename 옵션은 TFTP로 가져오려는 파일의 이름입니다.

dhcpd 설정 파일을 편집한 다음에, **/etc/init.d/isc-dhcp-server restart** 명령으로 **dhcpd**를 다시 시작하십시오.

4.3.3 BOOTP 서버 준비하기

GNU/리눅스용 BOOTP 서버는 두 가지가 있습니다. 하나는 CMU **bootpd**이고, 다른 프로그램은 실제로 DHCP서버인, ISC **dhcpd**가 있습니다. 이 프로그램은 bootp 와 isc-dhcp-server라는 패키지로 포함되어 있습니다.

CMU **bootpd**를 사용하려면 먼저 /etc/inetd.conf 파일의 해당 라인의 주석을 제거(또는 추가)해야 합니다. 데비안 GNU/리눅스에서는 **update-inetd --enable bootps** 명령을 실행하고 이어 **/etc/init.d/inetd reload** 명령을 실행합니다. BOOTP 서버가 데비안을 실행하지 않는 경우, 문제의 줄은 다음과 같이 합니다:

```
bootps  dgram  udp  wait  root  /usr/sbin/bootpd  bootpd -i -t 120
```

여기서 /etc/bootptab 파일을 만듭니다. 이 파일 형식은 오래전 BSD 방식의 printcap, termcap, disktab 파일처럼 친숙하면서도 알기 어려운 형식으로 되어 있습니다. 자세한 내용은 bootptab 메뉴얼 페이지를 보십시오. CMU **bootpd** 명령은 클라이언트의 하드웨어 (MAC) 주소를 미리 알아둬야 합니다. 다음은 /etc/bootptab의 예입니다:

```
client:\
  hd=/tftpboot:\
  bf=tftpboot.img:\
  ip=192.168.1.90:\
```

```
  sm=255.255.255.0:\
  sa=192.168.1.1:\
  ha=0123456789AB:
```

적어도 클라이언트의 하드웨어 주소를 지정하는 "ha" 옵션은 변경해야 합니다. "bf" 옵션은 TFTP로 클라이언트가 받게 될 파일을 지정합니다. 자세한 내용은 4.3.5절 부분을 참조하십시오.

반대로, ISC **dhcpd** 설정은 정말 쉽습니다. BOOTP 클라이언트를 DHCP 클라이언트의 일종의 특별한 케이스로 취급하기 때문입니다. 일부 아키텍쳐에서는 BOOTP로 부팅하게 하려면 복잡한 설정이 필요합니다. 그러한 아키텍처의 경우는 4.3.2절 부분을 참고하십시오. 그러한 경우가 아니라면, `/etc/dhcp/dhcpd.conf` 파일에서 **`allow bootp`**를 클라이언트가 들어 있는 서브넷의 설정 부분에 집어 넣고, **`/etc/init.d/isc-dhcp-server restart`** 명령으로 **dhcpd**를 다시 시작하면 됩니다.

4.3.4 TFTP 서버 사용하기

이동 TFTP 서버를 준비하려면, 먼저 **tftpd**이 활성화되어 있는지 확인해야 합니다.

`tftpd-hpa`의 경우에 서비스가 실행 될 수 있는 2가지 방법이 있습니다. 그것은 시스템의 `inetd` 데몬에서 요구에 따라 시작하거나 독립 데몬으로 실행되도록 설정할 수 있습니다. 이러한 방법 중에 어떤 패키지를 다시 구성하여 설치할 때 사용되며 변경할 수 있습니다.

> 참고
>
> 
>
> 전통적으로 부팅 이미지를 저장하는 위치로 TFTP 서버는 `/tftpboot` 디렉터리를 사용합니다. 하지만, 데비안 GNU/리눅스 패키지는 Filesystem Hierarchy Standard에 맞추려고 다른 디렉터리를 사용합니다. 예를 들어, `tftpd-hpa`는 기본값으로 `/srv/tftp`를 사용합니다. 여기서 설명하는 설정 파일 예제에서는 이 점을 고려해서 적용하십시오.

데비안에서 사용할 수 있는 모든 **in.tftpd** 대체품은 기본적으로 시스템 로그에 TFTP 요청을 기록해야 합니다. 그 중에서는 **`-v`** 옵션을 사용하면 더 많이 기록합니다. 부팅 문제가 발생한 경우 이 로그 메시지를 확인해 보십시오. 오류의 원인을 파악하는데 좋은 출발점입니다.

4.3.5 TFTP 이미지를 적당한 위치에 놓기

그 다음에, 필요한 TFTP 부팅 이미지를 4.2.1절에 쓰여 있는 것처럼 **tftpd** 부팅 이미지 디렉터리에 복사해 놓으십시오. 이 파일에서, **tftpd**가 특정 클라이언트를 부팅할 때 사용하는 특정 파일로 링크를 만들어야 할 것입니다. 불행히도 그 파일의 이름은 TFTP 클라이언트가 결정하고, 어떤 표준도 없습니다.

4.4 자동 설치

여러 컴퓨터에 설치하려면 완전 자동 설치가 가능합니다. 이것을 위한 데비안 패키지는 `fai-quickstart`와 (설치 서버로 사용 가능) 데비안 설치 자체를 포함합니다. 자세한 내용은 FAI 홈페이지를 참조하십시오.

4.4.1 데비안 설치 프로그램을 이용한 자동 설치

데비안 설치프로그램은 preconfiguration 파일을 이용해 자동 설치를 지원하고 있습니다. preconfiguration 파일은 네트워크 또는 이동식 미디어로 읽어 설치 중에 질문에 대한 답변을 묻어가는 데 사용됩니다.

부록 B에 보면 미리 설정에 대한 문서가 있고, 약간 고쳐서 사용할 수 있는 잘 동작하는 예제가 있습니다.

4.5 설치 파일의 무결성 확인하기

데비안 미러에 제공된 SHA256SUMS 또는 SHA512SUMS 체크섬을 이용해 다운로드한 파일의 무결성을 확인할 수 있습니다. 체크섬 파일은 해당 설치 이미지와 같은 위치에 있습니다. 다음 위치를 보십시오:

- CD 이미지의 체크섬 파일,
- DVD 이미지의 체크섬 파일,
- 기타 설치 파일의 체크섬 파일.

다운로드한 설치 파이르이 체크섬을 계산하려면, 다음을 사용하십시오:

```
sha256sum filename.iso
```

및

```
sha512sum filename.iso
```

그리고 표시된 체크섬을 SHA256SUMS 및 SHA512SUMS 파일의 체크섬과 비교하십시오.

Debian CD FAQ에 이 주제에 대해 (예를 들어 위 과정을 반자동 실행하는 check_debian_iso 스크립트) 더 유용한 정보가 있습니다. 또 위 체크섬 파일 자체의 무결성을 검사하는 방법도 있습니다.

제 5 장

설치 시스템 부팅하기

5.1 32-bit hard-float ARMv7에서 설치 프로그램 부팅하기

5.1.1 부팅 이미지 형식

ARM 시스템에서는 대부분의 경우 1가지나 2가지의 부팅 이미지 형식을 사용합니다. (1) 표준 리눅스 zImage 형식의 커널 (“vmlinuz”) 및 표준 리눅스 초기 램디스크 (“initrd.gz”), (2) uImage 형식의 커널 (“uImage”) 및 거기 해당하는 최초 램디스크 (“uInitrd”).

uImage/uInitrd는 U-Boot 펌웨어에서 사용하려고 만들어진 이미지 형식입니다. u-boot는 여러 ARM 시스템에서 (주로 32비트 시스템) 사용합니다. 예전 버전의 U-Boot에서는 uImage/uInitrd 형식의 파일만 부팅할 수 있습니다. 즉 이 형식은 예전의 armel 시스템에서 주로 사용합니다. 최근 버전의 U-Boot에서는 uImage/uInitrd 부팅 말고 표준 리눅스 커널과 램디스크 이미지로 부팅할 수 있습니다. 하지만 uImage 부팅과는 명령어 문법이 약간 다릅니다.

멀티 플랫폼 커널을 사용하는 시스템에서는, 커널과 최초 램디스크 외에 디바이스-트리 파일 (device-tree blob, “DTB”라고도 합니다) 이 필요합니다. 이 파일은 지원하는 시스템마다 다르고, 특정 하드웨어에 대한 설정이 들어 있습니다. DTB는 시스템의 펌웨어에서 만들어 내지만, 최근 시스템에서는 보통 따로 읽어들여야 합니다.

5.1.2 콘솔 설정

네트워크 부팅 타르볼 (5.1.3.2절) 과 설치 SD 카드 이미지 (5.1.5절) 는 U-Boot의 “console” 변수에서 정의한 (플랫폼 전용) 기본 콘솔을 사용합니다. 대부분의 경우 이 콘솔은 시리얼 콘솔이므로, 이 플랫폼에서는 설치 프로그램 사용에 시리얼 콘솔 케이블이 필요합니다.

비디오 콘솔을 지원하는 플랫폼의 경우, 설치 프로그램을 비디오 콘솔에서 시작하고 싶으면 U-Boot “console” 변수를 적당히 수정할 수 있습니다.

5.1.3 TFTP로 부팅하기

네트워크에서 부팅하려면, 네트워크에 연결되어 있어야 하고 TFTP 네트워크 부팅 서버가 (그리고 네트워크 자동 설정에 필요한 DHCP, RARP 혹은 BOOTP 서버가) 필요합니다.

서버 쪽에서 네트워크 부팅을 설정하는 방법은 4.3절에 설명되어 있습니다.

5.1.3.1 U-Boot에서 TFTP 부팅

U-Boot 펌웨어를 사용하는 시스템에서 네트워크 부팅은 3가지 단계로 이루어져 있습니다: (1) 네트워크 설정, (2) 이미지(커널/최초 램디스크/DTB) 메모리에 읽어들이기, (3) 읽어들인 코드 실행.

먼저 네트워크를 설정해야 합니다. 다음을 실행해 DHCP로 자동 설정할 수 있습니다:

```
setenv autoload no
dhcp
```

아니면 수동으로 환경 변수를 설정할 수 있습니다:

```
setenv ipaddr <클라이언트의 IP 주소>
setenv netmask <네트마스크>
setenv serverip <TFTP 서버의 IP 주소>
setenv dnsip <네임서버의 IP 주소>
setenv gatewayip <기본 게이트웨이의 IP 주소>
```

위 설정을 저장하고 싶으면 다음과 같이 합니다:

```
saveenv
```

그 다음에 이미지(커널/최초 램디스크/DTB)를 메모리에 읽어들여야 합니다. TFTP 명령에 메모리를 읽어들일 위치의 주소를 써야 합니다. 하지만 메모리 배치가 시스템마다 다르기 때문에 어떤 주소를 사용해야 하는지는 일반적인 규칙은 없습니다.

일부 시스템에서는, U-Boot에 적합한 로딩 주소가 환경 변수로 미리 정의되어 있습니다: kernel_addr_r, ramdisk_addr_r 및 fdt_addr_r. 이 환경 변수가 정의되어 있는지 여부를 다음 명령으로 확인해 볼 수 있습니다

```
printenv kernel_addr_r ramdisk_addr_r fdt_addr_r
```

이 값이 정의되어 있지 않으면, 시스템의 문서에서 적절한 값을 확인해 보고 직접 값으 지정해야 합니다. 예를 들어 Allwinner SunXi SOC 기반 시스템(예: Allwinner A10, 아키텍쳐 이름 “sun4i” 또는 Allwinner A20, 아키텍쳐 이름 “sun7i”)의 경우, 다음 값을 사용합니다.

```
setenv kernel_addr_r 0x46000000
setenv fdt_addr_r 0x47000000
setenv ramdisk_addr_r 0x48000000
```

로딩 주소를 지정하면, 다음과 같이 앞에서 지정한 TFTP 서버에서 이미지를 메모리에 읽어들일 수 있습니다:

```
tftpboot ${kernel_addr_r} <커널 이미지 파일 이름>
tftpboot ${fdt_addr_r} <DTB 파일 이름>
tftpboot ${ramdisk_addr_r} <최초 램디스크 이미지 파일 이름>
```

3번째는 커널 커맨드라인을 설정하고 읽어들인 코드를 실행하는 부분입니다. u-boot는 “bootargs” 환경 변수의 내용을 커널의 커맨드라인으로 넘깁니다. 그러므로 커널 및 설치 프로그램의 파라미터는(콘솔 장치(5.3.1절 참고) 또는 미리 설정 옵션(5.3.2절 및 부록 B 참고)) 다음과 같은 명령으로 설정할 수 있습니다:

```
setenv bootargs console=ttyS0,115200 rootwait panic=10
```

읽어들인 코드를 실행하는 정확한 명령은 이미지 형식에 따라 다릅니다. uImage/uInitrd의 경우 명령어는 다음과 같고,

```
bootm ${kernel_addr_r} ${ramdisk_addr_r} ${fdt_addr_r}
```

네이티브 리눅스 이미지의 경우 다음과 같습니다:

```
bootz ${kernel_addr_r} ${ramdisk_addr_r}:${filesize} ${fdt_addr_r}
```

표준 리눅스 이미지로 부팅할 때, 커널과 DTB를 읽어들이고 다음에 최초 램디스크 이미지를 읽어들이는 게 중요합니다. U-Boot에서는 파일 크기 변수를 마지막에 읽어들인 파일의 크기로 설정하고, bootz 명령이 제대로 동작하려면 램디스크 이미지의 크기가 필요하기 때문입니다. 플랫폼 전용 커널로 부팅하는 경우(예를 들어 디바이스 트리 없는 커널)에는 ${fdt_addr_r} 파라미터를 생략하면 됩니다.

5.1.3.2 미리 빌드된 네트워크 부팅 타르볼

데비안에는 TFTP 서버에 풀어 놓을 수 있는, 미리 빌드된 네트워크 부팅 타르볼이(.../images/netboot/-netboot.tar.gz) 들어 있고 여기에는 네트워크 부팅에 필요한 모든 파일이 들어 있습니다. 또 설치 프로그램을 읽어들이는 부팅 스크립트도 들어 있습니다. 최근의 U-Boot 버전에는 TFTP 자동 부팅 기능이 들어 있어서 부팅 가능한 로컬 저장 장치가(MMC/SD, USB, IDE/SATA/SCSI) 없으면 TFTP 자동 부팅을 시도하고 TFTP 서버에서 이 부팅 스크립트를 읽어들입니다. 이 기능을 사용하려면 먼저 네트워크에 DHCP 서버가 있어야 하고 이 서버에서 DHCP 클라이언트 쪽에 TFTP 서버의 주소를 안내해야 합니다.

U-Boot 명령행에서 TFTP 자동 부팅 기능을 사용하려면 다음 명령을 사용할 수 있습니다:

```
run bootcmd_dhcp
```

그렇게 하지 않고 수동으로 타르볼의 부팅 스크립트를 읽어들이려면, U-Boot 프롬프트에서 다음 명령을 실행할 수 있습니다:

```
setenv autoload no
dhcp
tftpboot ${scriptaddr} /debian-installer/armhf/tftpboot.scr
source ${scriptaddr}
```

5.1.4 U-Boot 이용해 USB 메모리에서 부팅하기

최근의 U-Boot 버전에서는 USB를 지원하므로, USB 메모리와 같은 USB 대용량 저장소 장치에서 부팅할 수 있습니다. 아쉽지만 부팅하는 정확한 단계는 하드웨어마다 조금씩 다를 수 있습니다.

U-Boot v2014.10 버전부터 공통된 명령행 처리와 자동 부팅 프레임워크가 갖춰졌습니다. 이 기능 때문에 이 프레임워크를 구현한 시스템이라면 어디든 동작하는 일반적인 부팅 이미지를 만들 수 있게 되었습니다. `debian-installer`에서는 USB 메모리를 이용해 그러한 시스템을 부팅할 수 있습니다. 하지만 이 프레임워크를 아직 사용하지 않는 플랫폼도 있습니다.

데비안 설치에 사용할 부팅 가능 USB 메모리를 만드려면, hd-media 묶음을 (4.2.1절 참고) USB 메모리에 풀어 놓습니다. USB 메모리는 하드웨어의 U-Boot 버전에서 지원하는 파일 시스템으로 포맷해야 합니다. 최근의 U-Boot 버전에서는 FAT16 / FAT32 / ext2 / ext3 / ext4 모두 동작합니다. 그리고 첫번째 데비안 설치 CD 또는 DVD의 ISO 이미지를 그 USB 메모리에 복사합니다.

최근 U-Boot 버전의 자동 부팅 프레임워크는 PC BIOS/UEFI의 부팅 순서와 비슷하게 동작합니다. 즉 가능한 부팅 장치에서 부팅 이미지를 확인하고, 찾은 부팅 장치 중에서 첫 번째에서 부팅합니다. 운영 체제를

설치하지 않았으면, USB 메모리를 연결하고 전원을 켜면 설치 프로그램을 시작하게 됩니다. U-Boot 프롬프트에서 “run usb_boot” 명령을 입력하면 언제든지 USB 부팅을 할 수 있습니다.

시리얼 콘솔을 사용할 때 USB 메모리에서 부팅할 경우 발생할 수 있는 한 가지 문제는 보우레이트가 일치하지 않는 경우입니다. 콘솔 변수를 U-Boot에서 지정한 경우, `debian-installer` 부팅 스크립트에서 자동으로 커널에 전달해서 사용하는 콘솔 장치와 보우레이트를 설정합니다. 아쉽지만 콘솔 변수 처리는 플랫폼마다 다릅니다. 일부 시스템에서는 콘솔 변수에서 보우레이트를 지정하고 (“console=ttyS0,115200”처럼), 어떤 플랫폼에서는 콘솔 변수에 시리얼 장치만 (“console=ttyS0”처럼) 설정합니다. 후자의 경우 U-Boot와 커널이 사용하는 기본 보우레이트가 다른 경우 콘솔 출력이 깨질 수 있습니다. 최근의 u-boot 버전에서는 115200 보우레이트를 사용하지만, 여전히 커널은 전통적인 9600 보우레이트를 사용합니다. 이런 상황이 벌어지는 경우, 콘솔 변수를 수동으로 설정해서 시스템의 보우레이트를 바로잡고 “run usb_boot” 명령으로 설치 프로그램을 시작해야 합니다.

5.1.5 설치 프로그램에서 빌드한 SD 카드 이미지 사용하기

여러가지 시스템에 대해, 데비안에는 U-Boot와 `debian-installer`가 모두 들어 있는 SD 카드 이미지가 있습니다. 이 이미지는 두 가지 종류로 나옵니다. 하나는 네트워크에서 소프트웨어 패키지를 다운로드하는 이미지이고 (.../images/netboot/SD-card-images/ 위치에 있습니다) 또 하나는 데비안 CD/DVD를 사용하는 오프라인 설치 이미지입니다(.../images/hd-media/SD-card-images/ 위치에 있습니다). 저장 공간과 네트워크 사용량을 절약하려고 이미지는 2가지 부분으로 구성되어 있습니다. 하나는 시스템 의존적인 부분으로 파일 이름이 “firmware.<시스템종류>.img.gz”이고, 또 하나는 시스템 독립적인 부분으로 파일 이름이 “partition.img.gz”입니다.

이 2가지 부분을 리눅스 시스템에 필요한 완전한 이미지로 만드려면 다음과 같이 zcat을 사용합니다:

```
zcat firmware.<시스템종류>.img.gz partition.img.gz > complete_image.img
```

윈도우 시스템에서는 먼저 이 2가지 부분을 각각 압축을 풀어야 합니다. 압축은 7-Zip과 같은 프로그램으로 풀 수 있습니다. 그리고 압축을 푼 이 2가지 부분을 합칩니다. 윈도우 CMD.exe 창에서

```
copy /b firmware.<system-type>.img + partition.img complete_image.img
```

명령으로 합칠 수 있습니다.

최종 이미지를 SD 카드에 씁니다. 예를 들어 리눅스 시스템에서는 다음 명령을 실행하면 됩니다:

```
cat complete_image.img > /dev/SD카드장치
```

SD 카드를 타겟 시스템에 꽂고 시스템 전원을 넣으면 SD 카드에서 설치 프로그램을 읽어들입니다. 오프라인 설치를 위해 hd-media 설치 방법을 사용하려면, 설치 프로그램에 첫번째 데비안 CD/DVD를 별도의 미디어로 접근하는 방법을 (예를 들어 USB 메모리의 CD/DVD ISO 이미지로 접근하는 방법) 설정해야 합니다.

설치 프로그램의 파티션 나누기 단계에서(6.3.4절 참고), 카드의 파티션을 삭제하거나 바꿀 수 있습니다. 일단 설치 프로그램이 시작하면 시스템의 메인 메모리에서 동작하기 때문에 SD 카드에 접근할 필요가 없어집니다. 그러므로 카드 전체를 데비안 설치에 사용할 수 있습니다. SD 카드에 적절한 파티션을 만드는 가장 쉬운 방법은, 설치 프로그램에서 자동으로 만드는 방법입니다(6.3.4.2절 참고).

5.2 접근성

눈이 불편하다는 등의 이유로 특별한 지원이 필요한 사용자가 있습니다. 많은 접근성 기능은 수동으로 활성화해야합니다. 접근성 기능을 사용하는 부팅 파라미터를 추가할 수 있습니다. 대부분의 아키텍처는 키보드를

QWERTY 키보드로 인식하니 주의하십시오.

5.2.1 설치 프로그램 프론트엔드

데비안 설치 프로그램은 사용자에게 질문하고 답변하는 인터페이스로 여러가지 프론트엔드를 지원합니다. 접근 편의에 따라 여러가지가 있습니다: **text**는 일반 텍스트 인터페이스이고, **newt**는 텍스트를 사용하는 대화상자를 사용합니다. 부팅 명령으로 프론트엔드를 선택할 수 있습니다. 5.3.2절의 **DEBIAN_FRONTEND** 문서를 참고하십시오.

newt 프론트엔드에서는 (점자와 같이 사용), 보통 화살표 키로 답을 선택하고 **Enter**를 눌러 그 선택을 확인합니다. **Tab** 또는 **Shift** - **Tab** 키를 누르면 대화창 요소 사이를 전환합니다. 특히 **뒤로 이동** 단추로 이동하는데 사용합니다. 이 단추를 누르면 이전 질문으로 돌아갑니다. 일부 대화창에는 확인란이 있는데, **Space** 키를 눌러 켜거나 끌 수 있습니다.

newt 프론트엔드에서는 (음성과 같이 사용), 보통 숫자와 **Enter** 키를 눌러 답을 선택합니다. 또 아무것도 입력하지 않고 **Enter** 키를 누르면 기본값을 받아들입니다. **<** 문자와 **Enter** 키를 누르면 이전 질문으로 돌아갑니다. 여러 개를 선택해야 하는 경우 (예를 들어, 태스크 선택), **!** 문자를 입력해 아무 것도 선택하지 않음을 표현할 수 있습니다.

5.2.2 보드 장치

일부 접근성 장치는 컴퓨터 내부에 연결하는 보드와 비디오 메모리에서 직접 텍스트를 읽습니다. 작동에는 **fb=false** 부팅 파라미터를 사용하여 프레임 버퍼 지원을 비활성화 해야합니다 . 그러나 이것은 사용 가능한 언어가 줄어 듭니다.

5.2.3 고대비 테마

시력이 좋지 않은 사용자의 경우 설치 프로그램에서 고대비 색상 테마를 사용하면 더 읽기 좋습니다. 이 기능을 사용하려면 부팅 화면에서 “Accessible high contrast” 항목을 사용합니다. **d** 단축키를 누르거나 부팅 파라미터에 **theme=dark** 파라미터를 사용할 수 있습니다.

5.2.4 화면 확대

시력이 낮은 사용자를 위해, 그래픽 설치에서는 기본적인 화면 확대 기능이 들어 있습니다. Control-+ 및 Control– 단축키를 눌러 글꼴 크기를 늘리거나 줄일 수 있습니다.

5.2.5 전문가 설치, 복구 모드, 자동 설치

전문가 설치, 복구 모드, 자동 설치도 접근성 기능에서 쓸 수 있습니다. 이 기능을 사용하려면, 먼저 **a**를 입력해 “고급 옵션” 하위 메뉴를 들어갑니다. BIOS 시스템을 사용하는 경우 (부팅 메뉴가 한 번만 삑 소리를 냅니다), 뒤에 **Enter** 키를 누릅니다. UEFI 시스템의 경우 (부팅 메뉴가 두 번 소리를 냅니다) **a**를 누르기만 하면 됩니다. 그 다음 음성 합성 기능을 사용하고 싶으면, **s**를 누릅니다 (마찬가지로 BIOS 시스템에서는 **Enter** 키를 누르고 UEFI 시스템에서는 하지 않습니다). **x**는 전문가 설치, **r**은 복구 모드, **a**는 자동 설치입니다. 마찬가지로 BIOS 시스템을 사용하면 뒤에 **Enter** 키를 누릅니다.

자동 설치의 경우 미리 설정 기능을 사용해 데비안을 완전히 자동으로 설치할 수 있습니다. 미리 설정을 가져올 위치는 접근성 기능이 시작한 뒤에 입력할 수 있습니다. 미리 설정 기능은 부록 B에 설명되어 있습니다.

5.2.6 설치한 시스템의 접근성

설치한 시스템의 접근성에 대한 문서는 데비안 접근성 위키 페이지에 있습니다.

5.3 부팅 파라미터

부팅 파라미터는 리눅스 커널 파라미터로 보통 주변 장치를 제대로 동작하도록 제어하는 데 이용합니다. 대부분 커널에서 주변 장치에 관한 정보를 자동으로 찾아 냅니다. 하지만 몇몇 경우에는 파라미터로 커널을 조금 도와줘야 합니다.

시스템을 첫번째로 부팅하는 경우라면, 기본 부팅 파라미터를 시도해 보시고(즉 파라미터를 사용하지 않는 것) 제대로 동작하는 지 보십시오. 보통은 제대로 동작합니다. 제대로 동작하지 않는 경우에 다시 부팅해서 하드웨어에 필요한 특별한 파라미터가 있는 지 찾아 보십시오.

부팅 파라미터에 관한 정보는 Linux BootPrompt HOWTO에(여러가지 보기드문 하드웨어에 대한 정보 포함) 있습니다. 여기서는 몇 가지 많이 이용하는 파라미터에 대한 대략만 다룹니다. 많이 발생하는 문제점 몇 개는 5.4절에 들어 있습니다.

5.3.1 부팅 콘솔

시리얼 콘솔로 부팅하는 경우, 보통 자동으로 찾아냅니다. 시리얼 콘솔로 부팅하려는 컴퓨터에 비디오 카드와 (프레임버퍼) 키보드가 붙어 있다면, **console=장치** 파라미터를 커널에 넘겨야 할 수 있습니다. 여기서 장치는 해당 시리얼 장치이고, ttyS0과 같이 씁니다.

속도와 패리티 등의 시리얼 포트 파라미터를 지정해야 합니다. 예를 들어 **console=ttyS0,9600n8**와 같이 합니다. 57600이나 115200도 많이 사용하는 속도입니다. "---" 다음에 이 옵션을 지정하십시오. 그래야 (부트로더 설치 모듈에서 지원하는 경우) 설치한 시스템의 부트로더 설정으로 들어갑니다.

설치 프로그램이 사용하는 터미널 종류가 사용하고 있는 터미널 에뮬레이터에 맞도록 하려면, **TERM=종류** 파라미터를 추가할 수 있습니다. 단 설치 프로그램은 다음 터미널 종류만 지원합니다: linux, bterm, ansi, vt102, dumb. debian-installer의 시리얼 콘솔의 기본값은 **vt102**입니다. IMPI 콘솔이나 기타 이러한 터미널 종류를 지원하지 않는 가상화 도구를 사용하고 있다면(예를 들어 QEMU/KVM), **screen** 세션 안에서 그 툴을 시작하면 됩니다. 그러면 screen 터미널 타입으로 동작하는데 vt102 터미널과 매우 가깝습니다.

5.3.2 데비안 설치프로그램 파라미터

설치 시스템에서는 유용하게 쓸 수도 있는 부팅 파라미터 몇 개를 더 인식합니다[1].

여러가지 파라미터는 "짧은 형식"이 있어서 커널 명령행의 길이 제한을 피하면서 쉽게 입력할 수 있습니다. 어떤 파라미터에 짧은 형식이 있는 경우에는, (일반적인) 긴 형식 뒤에 괄호 안에 써 놓았습니다. 이 안내서에 들어 있는 예제에서도 짧은 형식을 사용합니다.

debconf/priority (priority) 이 파라미터는 표시할 메시지의 가장 낮은 우선순위를 설정합니다.

> 기본 설치에서는 **priority=high**를 사용합니다. 즉, 높음 우선 순위와 필수 우선순위의 메시지를 표시하지만, 보통과 낮음 우선 순위 메시지는 넘어갑니다. 문제가 발생할 경우에는 설치 프로그램에서 우선순위를 필요에 따라 조정합니다.

[1] 현재 커널에서는 (2.6.9 이후) 32개의 명령행 옵션과 32개의 환경변수를 사용할 수 있습니다. 이 개수를 넘어가면 커널은 멎어버립니다. 또 전체 명령행 길이는 255자까지만 쓸 수 있고, 넘어가면 아무런 안내 없이 잘립니다.

priority=medium을 부팅 파라미터로 쓴 경우, 설치 메뉴에서 설치 메뉴에서 더 많은 조정을 할 수 있습니다. **priority=low**라고 쓴 경우, 모든 메시지를 볼 수 있습니다. (expert 부팅 방법과 동일합니다.) **priority=critical**의 경우, 필수 메시지만 표시하고 질문에 신경 쓰지 않고 설치를 진행합니다.

DEBIAN_FRONTEND 이 부팅 파라미터는 설치 프로그램에서 사용할 사용자 인터페이스 종류를 설정합니다. 현재 가능한 파라미터 값은:

- **DEBIAN_FRONTEND=noninteractive**
- **DEBIAN_FRONTEND=text**
- **DEBIAN_FRONTEND=newt**
- **DEBIAN_FRONTEND=gtk**

기본 프론트엔드는 **DEBIAN_FRONTEND=newt**입니다. 시리얼 콘솔 설치의 경우 **DEBIAN_FRONTEND=text**가 더 좋을 수도 있습니다. 일부 특이한 설치 미디어에서는 제한된 개수의 프론트엔드만 들어 있지만, **newt** 및 **text** 프론트엔드는 대부부느이 설치 미디어에서 사용할 수 있습니다. 그래픽을 지원하는 아키텍처에서는, 그래픽 설치 프로그램은 **gtk** 프론트엔드를 사용합니다.

BOOT_DEBUG 이 부트 파라미터를 2로 설정하면 설치프로그램의 부팅 절차를 자세하게 기록합니다. 이 값을 3으로 지정하면 부트 프로세스의 전략적인 부분에서 디버깅 쉘을 동작합니다. (부팅을 계속하려면 이 쉘을 끝내십시오.)

BOOT_DEBUG=0 기본값입니다.

BOOT_DEBUG=1 보통때보다 많은 디버깅 정보.

BOOT_DEBUG=2 디버깅 정보 아주 많이.

BOOT_DEBUG=3 부팅 과정의 곳곳에서 쉘을 실행해서 자세히 디버깅을 할 수 있습니다. 부팅을 계속하려면 쉘을 나가면 됩니다.

log_host, log_port 설치 프로그램의 로그 메시지를 로컬 파일에도 저장하면서 지정한 호스트와 포트에서 실행 중인 원격 syslog에 보냅니다. 포트를 지정하지 않으면 표준 syslog 포트인 514번을 기본값으로 사용합니다.

lowmem 사용 가능한 메모리에 따라 저용량 메모리를 판단하는 기준을 설치 프로그램의 기본값보다 높게 만드는 데 사용합니다. 쓸 수 있는 값은 1과 2입니다. 6.3.1.1절 부분도 참고하십시오.

noshell 설치 프로그램에서 tty2 및 tty3에 쉘을 실행하지 않습니다. 물리적인 보안이 확보되지 않은 상태에서 설치하는 경우에 좋습니다.

debian-installer/framebuffer (fb) 어떤 아키텍처에서는 많은 언어로 설치를 하려면 커널 프레임 버퍼를 사용합니다. 프레임 버퍼가 문제가 있는 경우 **fb=false** 매개 변수로 이 기능을 비활성화 할 수 있습니다. bterm이나 bogl 관련된 오류 메시지, 검은 화면 또는 설치를 시작하고 몇 분 후에 멈추는 것은 문제의 증상입니다.

debian-installer/theme (theme) 테마는 설치 프로그램의 사용자 환경(색, 아이콘 등)을 어떻게 보여줄 것인가를 결정합니다. 프론트엔드에 따라 사용할 수 있는 테마가 달라집니다. 현재 newt와 gtk 프론트엔드에서는 (기본 외에) "다크(dark)"테마만 있습니다. 이 다크 테마는 시각 장애 사용자를 위해 디자인되엇습니다. 이 테마를 사용하려면 부팅할 때 **theme=dark**를 파라미터로 넘기면 됩니다. (부팅 메뉴에서 단축키 **d**를 누를 수도 있습니다.)

netcfg/disable_autoconfig 기본값으로 debian-installer는 IPv6 자동 설정 및 DHCP를 통해 네트워크 설정을 검색합니다. 검색이 성공하면 그 설정을 검사하거나 바꿀 기회는 없습니다. 자동 설정이 실패할 경우에만 수동 네트워크 셋업을 할 수 있습니다.

로컬 네트워크에 IPv6 라우터나 DHCP 서버가 있으면서 이용하고 싶지는 않을 경우(예를 들어 잘못된 응답을 한다든지 때문에), **netcfg/disable_autoconfig=true** 파라미터를 사용하면 네트워크의(v4와 v6 모두) 자동 설정을 막고 수동으로 정보를 입력할 수 있습니다.

hw-detect/start_pcmcia PCMCIA에 문제가 있을 때 **false**로 하면 PCMCIA 서비스를 시작하지 않습니다. 일부 노트북 컴퓨터에서 이와 관련해 문제가 발생합니다.

disk-detect/dmraid/enable (dmraid) 설치 프로그램에서 Serial ATA RAID(ATA RAID, BIOS RAID, fake RAID라고도 함)의 지원을 활성화하려면 **true**로 설정합니다. 이 기능은 아직 실험중인 것을 주의하십시오. 추가 정보는 데비안 Installer Wiki에 있습니다.

preseed/url (url) 미리 설정해 놓은 파일을 다운로드할 URL을 지정합니다. 이 파일을 이용해 설치를 자동화합니다. 4.4절 참고.

preseed/file (file) 미리 설정해 놓은 파일을 읽어들일 URL을 지정합니다. 이 파일을 이용해 자동 설치를 합니다. 4.4절 참고.

preseed/interactive 미리 설정을 했더라도 질문을 표시하려면 **true**로 설정하십시오. 미리 설정 파일을 테스트하거나 디버깅하는 데 좋습니다. 이 설정은 부팅 파라미터로 넘긴 파라미터에 대해서는 효과가 없으니 주의하십시오. 부팅 파라미터에 대해서는 특별한 문법을 따로 사용합니다. 자세한 정보는 B.5.2절 부분을 보십시오.

auto-install/enable (auto) 미리 설정이 가능하기 전에 물어보는 질문을 네트워크 설정 다음으로 미룹니다. 이 자동 설치 사용법에 대한 자세한 정보는 B.2.3절 부분을 보십시오.

finish-install/keep-consoles 시리얼 콘솔이나 관리 콘솔에서 설치하면, 일반 가상 콘솔은(VT1-VT6) /etc/inittab에서 막습니다. 이 파라미터를 **true**로 하면 가상 콘솔을 막지 않습니다.

cdrom-detect/eject 기본값으로 debian-installer에서는 다시 시작하기 전에 설치할 때 사용했던 광학 미디어를 자동으로 뺍니다. 시스템이 그러한 미디어에서 자동으로 부팅하는 경우가 아니라면 이럴 필요가 없을 수도 있습니다. 어떤 경우에는 이렇게 하지 않는 게 좋을 때도 있습니다. 예를 들어 해당 광학 드라이브가 미디어를 다시 집어넣지 못하는 경우나, 사용자가 그 자리에 없기 때문에 수동으로 미디어를 집어넣지 못하는 경우입니다. 보통 슬롯 방식 드라이브는 미디어를 자동으로 다시 집어넣지 못합니다.

자동으로 미디어를 꺼내지 않으려면 **false**로 설정합니다. 단 설치한 후에 시스템이 광학 드라이브에서 자동으로 부팅하지 않도록 하십시오.

base-installer/install-recommends (recommends) **잘못** 옵션을 설정하면 패키지 관리 시스템이 자동으로 설치하는 동안에 시스템에 대한 "Recommends"을 설치 하지 않도록 됩니다. 6.3.5절도 참조하십시오.

이 옵션을 사용하면 매우 간결한 시스템이 됩니다. 반면에 일반적으로는 있는 기능이 이 시스템에서는 없을 가능성이 높습니다. 원하는 기능을 사용하려면 추천 패키지의 일부를 수동으로 설치해야 할 수도 있습니다. 그러므로 이 옵션은 숙련된 사용자만 사용해야 합니다.

debian-installer/allow_unauthenticated 기본 설정으로 설치 프로그램에서 사용하는 저장소는 알려진 GPG 키를 이용해 인증할 수 있어야 합니다. 이 인증을 하지 않으려면 **true**로 설정하십시오. **경고: 보안상 문제가 될 수 있으므로, 권장하지 않습니다.**

rescue/enable 일반적인 설치를 하지 않고 복구 모드로 변경하려면 **true**로 설정하십시오. 8.6절 부분을 참고하십시오.

5.3.3 부팅 파라미터로 질문에 답하기

일부 예외를 제외하고, 설치 과정에서 물어보는 모든 질문을 부팅 프롬프트에서 설정할 수 있습니다. 하지만 이 기능은 특수한 상황에서만 쓸모가 있습니다. 이 기능을 사용하는 방법은 B.2.2절 부분을 참고하십시오. 몇 가지 예제가 아래에 있습니다.

debian-installer/language (language), debian-installer/country (country), debian-installer/locale (loca

설치 중이나 설치 후에 사용하는 언어 국가와 로캘을 지정하는 방법은 두가지가 있습니다.

첫 번째, 쉽게 매개 변수를 `locale`로 전달하는 것입니다. 언어 및 국가는 그 가치로부터 파생됩니다. 예를 들면 언어는 독일어와 국가는 스위스로 선택하는 방법은 **`locale=de_CH`** 사용한다(`de_CH.UTF-8`는 설치한 시스템의 기본 locale로 선정됩니다). 제한 언어는 국가 및 locale의 모든 가능한 조합 방법을 얻을 수 있습니다.

두 번째, 별도로 좀 더 쉬운 옵션은 **언어**와 **국가**를 지정하는 것입니다. 이 경우에는 **장소**를 선택하여 설치한 시스템에 대한 구체적인 기본 `locale`를 지정 추가 할 수 있습니다. 예: **`언어=en 국가=DE locale=en_GB.UTF-8`**.

anna/choose_modules (modules) 기본값으로는 읽어들이지 않는 설치 프로그램 컴포넌트를 자동으로 읽어들이는 데 사용합니다. 유용한 추가 컴포넌트는 `openssh-client-udeb` (설치 도중에 **scp**를 사용할 수 있습니다) 및 `ppp-udeb`(D.4절 참고)이 있습니다.

netcfg/disable_autoconfig IPv6 자동 설정 및 DHCP를 끄고 강제로 고정 네트워크 설정을 하려면 **true**로 설정하십시오.

mirror/protocol (protocol) 기본적으로 설치 프로그램은 데비안 미러에서 파일을 다운로드하는 http 프로토콜을 사용하여 일반적인 우선순위에서는 설치 중에 ftp로 변경할 수 없습니다. 이 매개 변수를 **ftp**로 설정하면 설치 프로그램에 ftp를 사용하도록 강제할 수 있습니다. 목록에서 ftp 미러를 선택할 수는 없으므로, 호스트 이름을 입력해야한다는 것에주의하십시오.

tasksel:tasksel/first (tasks) `kde-desktop` 태스크처럼 태스크 목록에 나타나지 않는 태스크를 선택할 때 사용합니다. 자세한 정보는 6.3.6.2절 참고.

5.3.4 커널 모듈에 파라미터 넘기기

드라이버를 커널 안에 컴파일해 넣었다면, 커널 문서에 쓰여 있는 대로 파라미터를 넘길 수 있습니다. 하지만 드라이버를 모듈로 컴파일했다면 설치한 시스템에서는 부팅할 때 커널 모듈을 약간 다르게 읽어들이기 때문에, 일반적으로 하는 것처럼 모듈에 파라미터를 넘길 수 없게 됩니다. 그 대신에 설치 프로그램이 지원하는 특정 문법을 사용해 해당 파라미터를 올바른 설정 파일에 저장해 모듈을 읽어들일 때 사용하도록 만들 수 있습니다. 이 파라미터는 설치한 시스템의 설정에도 자동으로 적용됩니다.

한편 모듈에 파라미터를 넘겨야 하는 경우는 매우 드뭅니다. 보통 커널에서 시스템에 있는 하드웨어를 자동으로 찾아내서 쓸만한 기본값을 적용해 놓습니다. 하지만 일부 경우에는 파라미터를 수동으로 지정해야 할 수도 있습니다.

모듈의 파라미터를 설정하는 문법은 다음과 같습니다:

```
모듈_이름.파라미터_이름=값
```

같은 모듈 혹은 여러 모듈에 여러 개의 파라미터를 넘기려면, 이 문법을 반복해서 쓰면 됩니다. 예를 들어 오래된 3Com 네트워크 인터페이스 카드에서 BNC (동축) 커넥터와 IRQ 10을 지정하려면, 다음과 같이 합니다:

```
3c509.xcvr=3 3c509.irq=10
```

5.3.5 커널 모듈 블랙리스트

경우에 따라서는 모듈을 블랙리스트에 올려서 커널이나 udev가 자동으로 읽어들이지 않게 만들어야 합니다. 이렇게 하는 이유의 한 가지는 특정 모듈이 해당 하드웨어에서 문제를 일으키는 경우입니다. 또 같은 장치에 대해서 두 개의 다른 드라이버가 있기도 합니다. 드라이버가 충돌하거나 잘못된 드라이버를 먼저 읽어들이면 해당 장치가 제대로 동작하지 않을 수 있습니다.

다음 문법으로 모듈을 블랙리스트에 올릴 수 있습니다: **모듈_이름.blacklist=yes**. 이렇게 하면 해당 모듈을 `/etc/modprobe.d/blacklist.local`의 블랙리스트에 넣어서 설치 프로그램이나 설치한 시스템에 적용합니다.

설치 시스템이 모듈을 직접 읽어들이기도 합니다. 전문가 모드로 설치 프로그램을 시작해서 하드웨어 검색 단계에 나오는 모듈 목록에서 해당 모듈을 제외하면 모듈을 읽어들이지 않게 만들 수 있습니다.

5.4 설치 과정의 문제 해결

5.4.1 광학 미디어의 안정성

가끔, 특히 구형 드라이브의 경우, 광학 디스크에서 부팅이 실패할 수도 있습니다. 심지어는 (디스크에서 성공적으로 부팅한 경우에도) 디스크를 인식하지 못하거나 설치 도중에 디스크를 읽으면서 오류가 발생하는 경우도 있습니다.

이러한 문제는 여러가지 원인이 있을 수 있습니다. 여기서는 일부 많이 발생하는 문제 및 각각의 경우 대처하는 방법을 설명합니다. 나머지는 여러분에게 맡깁니다.

제일 먼저 확인할 수 있는 간단한 일이 두 가지 있습니다.

- 디스크가 부팅하지 않으면, 디스크를 올바르게 넣었는지 그리고 혹시 디스크 표면이 깨끗한지 확인하십시오.
- 설치 프로그램이 디스크를 인식하지 못하면, 설치 미디어 찾기 및 마운트 옵션을 다시 한 번 실행해 보십시오. 구형 CD-ROM 드라이브에서 발생하는 일부 DMA 관련 문제는 이런 식으로 해결된다고 알려져 있습니다.

이렇게 해도 해결되지 않으면, 아래에 나와 있는 방법을 시도해 보십시오. 전부는 아니지만 대부분의 경우, 여기에 나와 있는 방법은 CD-ROM과 DVD 모두에 적용됩니다.

광학 디스크에서 제대로 설치할 수 없는 경우, 사용할 수 있는 다른 설치 방법을 시도해 보십시오.

5.4.1.1 공통 사항

- 일부 오래 된 CD-ROM 드라이브는 최근의 CD 라이터에서 구운 디스크를 일지 못합니다.
- 일부 아주 오래 된 CD-ROM 드라이브는 "직접 메모리 접근"(DMA) 기능을 사용할 경우 올바르게 동작하지 않습니다.

5.4.1.2 문제점 파악 및 해결 방법

광학 디스크 부팅이 실패하면, 아래에 나온 방법을 시도해 보십시오.

- BIOS/UEFI가 광학 디스크 부팅을 지원하는 지 확인하고 (아주 오래 된 시스템이 경우에만 해당됩니다) 광학 드라이브가 해당 미디어를 지원하는 지 확인하십시오.
- ISO 이미지를 다운로드했다면, 그 이미지의 md5sum이 이미지를 받은 곳과 같은 곳에 있는 `MD5SUMS` 파일 안에 있는 md5sum과 일치하는 지 확인하십시오.

```
$ md5sum debian-testing-i386-netinst.iso
a20391b12f7ff22ef705cee4059c6b92  debian-testing-i386-netinst.iso
```

그 다음, 구운 디스크의 md5sum이 일치하는 지도 확인하십시오. 다음 명령을 사용하면 됩니다. 이미지의 크기를 이용해서 지정한 바이트 수만큼 디스크에서 읽어들입니다.

```
$ dd if=/dev/cdrom | \
> head -c `stat --format=%s debian-testing-i386-netinst.iso` | \
> md5sum
a20391b12f7ff22ef705cee4059c6b92  -
262668+0 records in
262668+0 records out
134486016 bytes (134 MB) copied, 97.474 seconds, 1.4 MB/s
```

설치 프로그램이 성공 후, 디스크를 찾을 수 없는 경우, 한 번 다시 시도하면 해결할 수 있습니다. 광학 드라이브가 여러 개 있는 경우 다른 광학 드라이브로 바꾸어보십시오. 그래도 동작하지 않거나 디스크를 인식하지만 읽으면서 오류가 발생하면 다음을 시도해 보십시오. Linux 기초 지식이 필요합니다. 명령을 실행하려면 먼저 두 번째 가상 콘솔(VT2)로 전환해서 쉘을 활성화하십시오.

- VT4로 전환해서 `/var/log/syslog`의 내용을 보고 (**nano**를 편집기로 사용) 특정 오류 메시지가 있는지 확인하십시오. 그 다음에 **dmesg**의 출력도 확인해 보십시오.
- **dmesg** 출력에서 광학 드라이브를 인식했는지 확인하십시오. 다음과 같은 내용이 있어야 합니다(연속된 줄이 아닐 수도 있습니다):

```
ata1.00: ATAPI: MATSHITADVD-RAM UJ-822S, 1.61, max UDMA/33
ata1.00: configured for UDMA/33
scsi 0:0:0:0: CD-ROM            MATSHITA DVD-RAM UJ-822S  1.61 PQ: 0 ANSI: 5
sr0: scsi3-mmc drive: 24x/24x writer dvd-ram cd/rw xa/form2 cdda tray
cdrom: Uniform CD-ROM driver Revision: 3.20
```

이와 같은 내용이 없으면, 드라이브가 연결된 컨트롤러를 인식하지 못했거나 아예 지원하지 않는다는 뜻입니다. 해당 컨트롤러에 무슨 드라이버가 필요한지 알려면, **modprobe** 명령을 사용해 수동으로 드라이버를 읽어들여 보십시오.

- /dev/ 아래에 광학 드라이브의 장치 노드가 있는 지 확인하십시오. 위의 예에서 장치 노드는 /dev/sr0 입니다. /dev/cdrom 파일도 있어야 합니다.
- **mount** 명령으로 광학 디스크가 이미 마운트되어 있지는 않은 지 확인하십시오. 마운트되어 있지 않다면 수동으로 마운트해 보십시오:

```
$ mount /dev/hdc /cdrom
```

  이 명령어 다음에 무슨 오류 메시지가 없는 지 확인하십시오.

- DMA가 켜져 있는 지 확인하십시오:

```
$ cd /proc/ide/hdc
$ grep using_dma settings
using_dma       1       0       1       rw
```

  첫번째 열의 using_dma 다음에 나오는 "1"은 DMA가 켜져 있다는 뜻입니다. DMA가 켜져 있다면 꺼 보십시오:

```
$ echo -n "using_dma:0" >settings
```

  광학 드라이브에 해당하는 장치 노드가 있는 디렉터리 안에서 실행해야 하는 것에 유의하십시오.

- 설치하는 데 문제가 있다면, 설치 프로그램의 메인 메뉴 맨 아래 부분에 있는 옵션을 사용해 설치 미디어가 올바른지 확인해 보십시오. 이 옵션은 디스크를 안정적으로 읽을 수 있는 지 시험하는 목적으로도 사용합니다.

5.4.2 부팅 설정

문제가 생겨서 커널이 부팅 과정에서 멈추거나, 주변 장치를 인식하지 못하거나, 드라이브를 제대로 인식하지 못하거나 하는 경우, 먼저 부팅 파라미터가 5.3절에 쓰여 있는 것처럼 제대로 되었는 지 확인하십시오.

장치의 펌웨어가 없어서 문제가 발생할 수도 있습니다. (2.2절 및 6.4절 참고.)

5.4.3 커널 시작 메시지 해석하기

부팅 과정에서, can't find something 혹은 something not present, can't initialize something, 아니면 심지어는 this driver release depends on something 형식의 메시지를 볼 수 있습니다. 이러한 메시지 대부분은 아무런 문제를 일으키지 않습니다. 설치 시스템은 여러 가지 주변 장치가 달린 컴퓨터에서 동작하도록 만들어졌기 때문에 이런 메시지가 나옵니다. 당연히 그 어떤 컴퓨터도 모든 종류의 주변 장치를 가진 컴퓨터는 없으므로, 운영체제에서는 찾으려는 주변 장치가 없을 때 이러한 메시지를 내보냅니다. 또 시스템이 일시적으로 멈추는 현상이 일어날 수도 있습니다. 이러한 현상은 어떤 장치가 응답할 때까지 기다리는데, 그 장치가 없을 때 발생합니다. 이런 시간이 너무 오래 걸린다고 생각한다면, 나중에 직접 설정한 커널을 사용할 수 있습니다(8.5절 참고).

5.4.4 설치 문제 보고하기

최초 부팅 단계를 지나갔지만 설치를 마치지 못했다면, 디버깅 기록 저장 메뉴가 도움이 될 수도 있습니다. 이 메뉴를 이용하면 시스템 오류 로그와 설정 정보를 설치 프로그램에서 저장 미디어로 복사하거나, 웹브라

우저를 이용해 다운로드할 수 있습니다. 이 정보는 무엇이 잘못되었는지 및 어떻게 고치는 지에 대한 단서가 들어 있습니다. 버그를 보고할 때 이 정보를 버그 보고에 첨부해 주십시오.

그 외의 설치 메시지는 설치할 때 /var/log/에 들어 있고, 설치된 시스템으로 부팅한 다음에는 /var/log/installer/에 들어 있습니다.

5.4.5 설치 보고 제출

그래도 문제가 있다면, 설치 리포트를 보내 주십시오. 설치가 성공했을 경우에도 설치 리포트를 보내 주시는 게 좋습니다. 그래야 사용자가 어떤 하드웨어 설정을 사용하는 지에 대한 정보를 많이 얻을 수 있습니다.

주의: 설치 보고서는 데비안 버그 추적 시스템(BTS)에 공개되며, 공개 메일링 리스트에도 전달됩니다. 공개해도 상관 없는 전자메일 주소를 사용하도록 하십시오.

작동하고있는 데비안 시스템이있다면, 설치 리포트를 보내는 가장 쉬운 방법은 다음과 같습니다. installation-report와 reportbug패키지를 설치(**apt install installation-report reportbug**)하고 ,8.4.2절에서 설명한대로 reportbug를 설정하여 **reportbug installation-reports**를 실행하십시오.

다른 방법으로, 설치 보고서를 작성하실 때 아래 형식을 이용하시고, installation-reports" 패키지에 대한 버그를 submit@bugs.debian.org로 메일을 보내 제출해 주십시오.

```
Package: installation-reports

Boot method: <설치 프로그램을 어떻게 부팅했는지? CD/DVD? USB 메모리? 네트워크?>
Image version: <설치 이미지를 받은 URL을 쓰는 게 가장 좋습니다>
Date: <설치한 날짜 및 시각>

Machine: <컴퓨터 설명(예, IBM Thinkpad R32)>
Processor:
Memory:
Partitions: >df -Tl 명령어의 결과. 파티션 테이블의 raw 정보가 좋습니다.>

Output of lspci -knn (or lspci -nn):

Base System Installation Checklist:
[O] = OK, [E] = Error (please elaborate below), [ ] = didn't try it

Initial boot:           [ ]
Detect network card:    [ ]
Configure network:      [ ]
Detect media:           [ ]
Load installer modules: [ ]
Detect hard drives:     [ ]
Partition hard drives:  [ ]
Install base system:    [ ]
Clock/timezone setup:   [ ]
User/password setup:    [ ]
Install tasks:          [ ]
Install boot loader:    [ ]
Overall install:        [ ]
```

```
Comments/Problems:

<설치 과정을 문장으로 설명하십시오. 그리고 처음에 설치했을 때
       들었던 생각, 평가, 아이디어 따위도 써 주십시오.>
```

버그 보고서에 문제가 무엇인지 설명하시고, 커널이 멈춘 경우에 마지막으로 볼 수 있는 커널 메시지를 넣으십시오. 문제가 발생할 때 어떤 과정을 거쳤는 지 설명하십시오.

제 6 장

데비안 설치 프로그램 사용하기

6.1 설치 프로그램이 동작하는 방식

이 아키텍쳐에서는 설치 프로그램에서는 텍스트 기반 인터페이스를 사용합니다. 현재 그래픽 인터페이스는 사용할 수 없습니다.

데비안 설치 프로그램은 여러 가지 용도의 컴포넌트로 구성되어 있고, 각 컴포넌트마다 설치 작업을 수행합니다. 각 컴포넌트는 설치 작업을 수행하면서 그 작업에 필요한 정보를 사용자에게 물어봅니다. 이 질문에는 우선순위가 부여되어 있고, 설치 프로그램이 맨 처음 시작할 때 물어볼 질문의 우선순위를 먼저 물어봅니다.

기본값으로 설치하면, 꼭 필요한(우선순위가 높은) 질문만 물어봅니다. 그래서 사용자가 거의 관여하지 않고 매우 자동적으로 설치할 수 있습니다. 컴포넌트는 순서대로 자동 실행합니다. 사용하는 설치 방법 및 하드웨어의 종류 따라 어떤 실행하는 컴포넌트가 달라집니다. 설치 프로그램에서 물어보지 않는 질문은 기본값을 사용합니다.

어떤 문제가 발생하면 오류 화면이 나타납니다. 그리고 설치 메뉴가 나타나서, 메뉴에서 앞의 문제를 피해 다른 작업을 선택할 수도 있습니다. 아무런 문제가 없으면 설치 메뉴를 볼 수 없고, 각 컴포넌트에 해당하는 질문에 차례대로 답을 하기만 하면 됩니다. 심각한 오류를 알리는 우선 순위는 '중요'이기 때문에 심각한 오류가 발생하면 반드시 오류 화면이 나타납니다.

설치 프로그램에서 사용하는 기본값 중에 몇 개는 `debian-installer`가 시작할 때 넘기는 부팅 파라미터에 따라 달라집니다. 예를 들어 강제로 고정 네트워크 설정을 하려면(사용할 수 있으면 IPv6 자동 설정 및 DHCP를 기본값으로 사용합니다) **`netcfg/disable_autoconfig=true`** 부팅 파라미터를 추가하면 됩니다. 사용할 수 있는 옵션에 대해서는 5.3.2절 부분을 참고하십시오.

고급 사용자라면 메뉴 방식 인터페이스가 더 편할 수도 있습니다. 메뉴 방식에서는 각 단계를 자동으로 진행하지 않고 사용자 입력에 따라 단계를 진행합니다. 설치 프로그램을 수동 메뉴 방식으로 사용하려면, **`priority=medium`** 파라미터를 사용하십시오.

커널 모듈을 설치하면서 파라미터를 넘겨야 하는 하드웨어에서는, "전문가" 모드로 설치 프로그램을 시작합니다. 설치 프로그램이 시작할 때 **expert** 명령을 사용하거나, **`priority=low`** 부팅 파라미터를 사용하면 됩니다. 전문가 모드에서는 `debian-installer`의 모든 부분을 마음대로 조정할 수 있습니다.

텍스트 기반 환경에서는 마우스 사용을 지원하지 않습니다. 여기서 대화 상자에서 왔다갔다할 때 쓰는 키를 설명합니다. **Tab** 혹은 **오른쪽** 화살표 키를 누르면 화면에 나오는 단추와 선택사항 중에서 "앞으로" 움직이고, Shift-Tab 혹은 **왼쪽** 화살표 키는 "뒤로" 움직입니다. **위쪽** 및 **아래쪽** 화살표 키를 누르면 스크롤 목록에서 선택할 항목을 움직이고, 스크롤 목록을 스크롤합니다. 또 긴 목록에서는, 글자를 하나 누르면 그

글자로 시작하는 항목이 있는 부분으로 직접 이동합니다. 또 **Pg-Up** 및 **Pg-Down**키로 목록을 스크롤합니다. **스페이스바**를 누르면 확인란 따위의 항목을 토글합니다. 선택한 항목으로 들어가려면 **Enter**를 누릅니다.

대화 창에는 추가 도움 정보가 있을 수도 있습니다. 도움말이 있는경우 화면의 맨 아래에 표시되고 **F1** 키를 눌러 볼 수 있습니다.

오류 메시지는 네번째 콘솔에서 나옵니다. 네번째 콘솔은 왼쪽 Alt-F4를 누르면(왼쪽 **Alt**를 누른 상태에서 **F4** 펑션 키) 볼 수 있습니다. 설치 메인 화면으로 돌아오려면 왼쪽 Alt-F1을 누릅니다.

이 메시지는 `/var/log/syslog` 파일에도 들어 있습니다. 설치한 후에는 이 로그는 새로 설치한 시스템의 `/var/log/installer/syslog` 파일로 복사됩니다. 그 밖에 설치할 때 나오는 메시지는 `/var/log/` 안에 들어 있고, 새로 설치한 시스템으로 부팅한 다음에 `/var/log/installer/` 안에 들어갑니다.

6.2 컴포넌트 소개

다음은 설치 프로그램의 컴포넌트와 각 컴포넌트가 하는 일에 대한 간단한 설명입니다. 특정 컴포넌트를 사용하는 방법에 대해 더 자세한 정보는 6.3절에 있습니다.

main-menu 설치 프로그램이 동작할 때 컴포넌트의 목록을 표시하고, 컴포넌트를 하나 선택하면 그 컴포넌트를 시작합니다. 메인 메뉴의 질문은 우선순위가 중간이기 때문에, 우선 순위를 높음이나 중요로 해 놓으면(기본값은 높음) 메인 메뉴를 볼 수 없습니다. 하지만 오류가 발생해서 사용자가 뭔가 작업을 해야 한다면, 사용자가 이 문제를 해결할 수 있도록 우선순위가 일시적으로 낮아지고, 이 경우 메인 메뉴가 나타날 수도 있습니다.

뒤로 가기 단추를 계속해서 눌러 현재 실행중인 컴포넌트를 나가게 되면 메인 메뉴로 갑니다.

localechooser 사용자가 설치 과정 및 설치할 시스템에서 사용할 지역화 옵션을(언어, 국가, 로캘) 선택합니다. 설치 프로그램에서는 선택한 언어로 메시지를 표시합니다. 단 그 언어로 번역이 다 되지 않았으면 일부 영어 메시지를 표시할 수도 있습니다.

console-setup 키보드 (레이아웃) 목록을 표시합니다. 여기에서 자기 키보드에 해당하는 모델을 선택합니다.

hw-detect 시스템의 하드웨어 대부분을 자동으로 검색합니다. 네트워크 카드, 디스크 드라이브, PCMCIA 등입니다.

cdrom-detect 데비안 설치 미디어를 찾아서 마운트합니다.

netcfg 인터넷을 통해 통신할 수 있도록 컴퓨터 네트워크 연결을 설정합니다.

iso-scan 하드 디스크에 들어 있는 ISO 이미지를 (`.iso` 파일) 찾습니다.

choose-mirror 데비안 아카이브 미러 목록을 표시합니다. 설치할 패키지가 들어 있는 위치를 선택합니다.

cdrom-checker 설치 미디어가 올바른지 확인합니다. 설치 이미지가 손상되지는 않았는지 직접 확인할 수 있습니다.

lowmem lowmem은 메모리가 작은 시스템을 검사하고, 여러가지 방법으로 `debian-installer`에서 필요 없는 부분을 메모리에서 없앱니다. (그 대신 일부 기능도 없어집니다.)

anna anna는 Anna's Not Nearly APT의 약자입니다. 선택한 미러 사이트 혹은 설치 미디어에서 가져온 패키지를 설치합니다.

user-setup 루트 암호를 설정하고 루트가 아닌 사용자를 추가합니다.

clock-setup 시스템 시계를 맞추고 시계를 UTC에 맞출 지 여부를 지정합니다.

tzsetup 앞에서 설정한 지역 정보에 따라 시간대를 설정합니다.

partman 시스템에 달린 디스크를 파티션하고, 파티션에 파일 시스템을 만들고, 파일 시스템을 마운트 위치에 마운트합니다. 완전 자동 모드 혹은 LVM 지원 기능과 같은 재미있는 기능도 들어 있습니다. 데비안 의 기본 파티션 도구입니다.

partitioner 여기서 시스템에 달린 디스크를 파티션합니다. 해당 컴퓨터 아키텍처에 적합한 파티션 프로그램을 이용합니다.

partconf 파티션 목록을 표시하고, 사용자 명령에 따라 선택한 파티션에 파일 시스템을 만듭니다.

partman-lvm 여기서 LVM(Logical Volume Manager)을 설정합니다.

partman-md 여기서 소프트웨어 RAID(Redundant Array of Inexpensive Disks)를 설정합니다. 이 소프트웨어 RAID는 최근 마더보드에 들어 있는 싸구려 IDE RAID (가짜 하드웨어 RAID) 컨트롤러보다는 대체로 우수합니다.

base-installer 다시 시작했을 때 데비안 GNU/리눅스가 동작하는 데 필요한 가장 기본적인 패키지를 설치합니다.

apt-setup APT를 설정합니다. 설치를 어떤 방식으로 하느냐에 따라 다르지만, 대부분을 자동으로 설정합니다.

pkgsel `tasksel` 프로그램을 이용해 소프트웨어를 추가로 선택하고 설치합니다.

os-prober 컴퓨터에 기존에 설치한 운영 체제를 찾아서 그 정보를 bootloader-installer에 넘깁니다. 그러면 bootloader-installer에서는 부트로더 시작 메뉴에 이 운영 체제를 추가합니다. 이렇게 하면 부팅할 때 어떤 운영 체제를 시작할 지 쉽게 선택할 수 있습니다.

bootloader-installer 여러가지 부트로더 설치 기능. 하드 디스크에 부트로더를 설치합니다. USB 메모리나 CD-ROM을 사용하지 않는 경우 Linux 사용해 컴퓨터를 시작하는데 필요합니다. 많은 부트로더에서는 부팅할 때마다 사용자가 어떤 운영 체제로 부팅할지 직접 선택할 수 있습니다.

shell 사용자가 메뉴에서 쉘을 실행하거나, 두번째 콘솔에서 쉘을 실행합니다.

save-logs 문제가 발생했을 때 관련 정보를 USB 메모리, 네트워크, 하드 디스크 등의 미디어에 기록합니다. 나중에 설치 프로그램의 소프트웨어 문제를 데비안 개발자에게 정확하게 알리는 데 이 기록을 이용합니다.

6.3 컴포넌트 사용하기

여기서는 설치 프로그램의 각 컴포넌트를 자세히 설명합니다. 이 컴포넌트는 사용자 입장에서 몇 단계로 분류할 수 있습니다. 여기서 설명하는 순서는 설치할 때 나타나는 순서입니다. 설치할 때 여기 있는 모듈을 모두 사용하지는 않습니다. 이 모듈 중에서 실제로 어떤 모듈을 사용하는 지는 설치 방법과 하드웨어에 따라 달라집니다.

6.3.1 데비안 설치 프로그램 준비 및 하드웨어 설정

데비안 설치 프로그램을 시작하여 첫 번째 화면이 표시되고 있다고 합시다. 이 때 debian-installer 기능은 아직 매우 제한적입니다. 하드웨어, 원하는 언어, 실행하는 작업 등에 대해서도 아직 모릅니다. 하지만 걱정하지 마십시오. debian-installer는 아주 똑똑하기 때문에, 하드웨어를 자동으로 검색해서, 필요한 구성 요소를 찾아내고, 고성능 설치 시스템으로 자신을 업그레이드 할 수 있습니다. 그러나(선호하는 언어, 키보드 배치, 사용할 네트워크 미러 선택처럼) 몇 가지 작업은 자동으로 알아낼 수 없으므로, debian-installer에게 알려줘야 합니다.

이 단계에서 debian-installer는 여러 번의 하드웨어 검색을 합니다. 첫번째는 설치 프로그램의 컴포넌트를 읽어들이는 데 필요한 하드웨어를(예를 들어 CD-ROM이나 네트워크 카드) 검색합니다. 첫번째로 검색할 때는 아직 사용할 수 없는 드라이버가 있기 때문에, 나중 단계에서 하드웨어 검색을 다시 합니다.

하드웨어 검색할 때 debian-installer에서 하드웨어 드라이버에서 펌웨어를 읽어들여야 하는 지 검사합니다. 펌웨어가 필요하지만 없는 경우에는, 없는 펌웨어를 이동식 미디어에서 읽어들일 수 있도록 대화상자를 표시합니다. 자세한 설명은 6.4절 부분을 참고하십시오.

6.3.1.1 사용 가능 메모리 검사 / 저용량 메모리 모드

debian-installer가 맨 처음에 하는 일 중의 하나는 메모리 검사입니다. 메모리가 부족할 경우 이 컴포넌트에서는 시스템에서 데비안 GNU/리눅스를 설치하는 데 문제가 없도록 설치 과정에 약간 수정을 가합니다.

설치 프로그램의 메모리 사용량을 줄이려고 첫째로 번역 기능을 사용하지 않습니다. 즉 설치는 영어로만 진행합니다. 물론 설치한 시스템은 설치가 끝난 다음에 지역화 기능을 설정할 수 있습니다.

그걸로 부족하다면, 설치 프로그램에서 기초적인 설치를 마칠 수 있는 컴포넌트만 읽어들이는 방법으로 메모리 사용량을 줄입니다. 이렇게 하면 설치 시스템의 기능이 줄어듭니다. 수동으로 컴포넌트를 추가로 읽어들일 수도 있지만, 컴포넌트를 하나 하나 선택할 때마다 메모리를 추가로 사용하기 때문에 설치가 실패할 수도 있습니다.

설치 프로그램이 저용량 메모리 모드로 동작하는 경우, 상당히 큰 스왑 파티션을(64MB– 128MB) 만들기를 권장합니다. 스왑 파티션을 가상 메모리로 사용해서 시스템에서 사용 가능한 메모리 양을 늘립니다. 설치 프로그램에서는 가능한 한 설치 과정의 앞 부분에서 스왑 파티션을 활성화합니다. 단 스왑 파티션을 자주 사용하면 시스템의 성능이 떨어지고 디스크 동작이 많아질 수 있습니다.

이렇게 하더라도 시스템의 메모리가 부족하면 시스템이 멈출 수도 있고, 예상하지 못한 오류가 발생하거나 커널이 프로세스를 끝낼 수도 있습니다. (이 경우에 “Out of memory” 메시지가 VT4와 syslog에 나타납니다.)

예를 들어 저용량 메모리 모드에서 스왑 공간이 부족하면 큰 EXT3 파일 시스템을 만들 때 실패한다는 보고가 있었습니다. 스왑을 더 늘려도 개선이 안 된다면, 파일 시스템을 EXT2로(EXT2는 설치 프로그램의 필수 컴포넌트입니다) 만들어 보십시오. 설치를 끝낸 다음에 EXT2 파티션을 EXT3로 바꿀 수 있습니다.

5.3.2절에 설명한 것처럼 “lowmem” 부팅 파라미터를 이용해, 설치 프로그램이 사용하는 저용량 메모리의 수준을 강제로 높일 수 있습니다. 사용 가능한 메모리로 자동 검색한 것보다 더 많은 메모리로 높일 수 있습니다.

6.3.1.2 지역화 옵션 선택

대부분의 경우 처음 물어 보는 질문은 설치할 때 및 설치한 시스템에 모두 사용할 지역화 옵션 선택에 대한 것입니다. 지역화 옵션은 언어, 위치, 로캘로 이루어져 있습니다.

여기서 선택한 언어를 사용해 나머지 설치 과정을 진행합니다. 단 그 언어로 해당 대화 상자의 번역이 있어야 됩니다. 그 언어로 된 번역문이 없으면 기본값인 영어를 사용합니다.

선택한 지리적 위치(대부분의 경우 국가)는 설치 과정의 뒷부분에서 기본 표준 시간대 추출과 그 나라에 적절한 데비안 미러를 고르는 데 이용합니다. 국가와 언어는 새 데비안 시스템의 로캘 결정이나 올바른 키보드 레이아웃 선택을 지원합니다.

먼저 사용할 언어를 선택합니다. 각 언어의 이름은 영어(왼쪽에) 및 해당 언어(오른쪽에)로 쓰여 있습니다. 오른쪽에 있는 이름은 해당 언어의 문자를 이용해서 표시됩니다. 이 언어 목록은 영어 이름 순서로 나열되어 있습니다. 목록의 맨 위에 "C" 로캘을 선택하는 옵션이 있습니다. "C" 로캘을 선택하면 설치는 영어로 진행하고, 설치한 시스템은 `locales` 패키지를 설치하지 않고 지역화 기능이 없게 됩니다.

다음은 지리적인 위치를 선택하라는 메시지가 있습니다. 언어 선택시 해당 언어가 여러 국가들에서 공식 언어로되어있는 경우 [1] 그 국가의 목록을 표시합니다. 목록에없는 국가를 선택하면 기타 (마지막 선택)을 선택하십시오. 그러면 대륙의 목록을 표시합니다. 대륙을 선택하면 관련 국가 목록을 표시합니다.

언어에 대해 국가가 하나 뿐이라면 국가 목록에 그 나라가 속한 대륙 또는 지역을 표시하고 그 나라를 기본적으로 선택합니다. 다른 대륙에있는 국가를 선택하고 싶은 경우 뒤로가기 를 선택하십시오.

참고

설치 시스템의 표준 시간대를 설정하면서 여러분 위치의 국가를 선택하는 일은 중요합니다.

지역이 정의되어 있지 않은 언어 및 국가의 조합을 선택하여 해당 언어에 여러 지역이 존재하는 경우, 그 중에서 설치한 시스템의 기본 지역을 선택하게 됩니다. [2] 그렇지 않으면 기본 지역은 선택한 언어 및 국가를 바탕으로 선택됩니다.

이전에 설명한 것처럼 선택된 기본 지역은 문자 코드 UTF-8 를 사용합니다.

낮은 우선순위로 설치하는 경우, 설치 시스템에 만들 로캘을 선택할 때 "레가시" 로캘을[3] 포함해 로캘을 선택합니다. 이 경우 어떤 로캘을 설치 시스템의 기본 로캘로 사용할지 묻습니다.

6.3.1.3 키보드 선택하기

어떤 키보드는 특정 언어에서 사용하는 문자에 맞게 만들어져 있습니다. 사용하고 있는 키보드에 맞는 키보드 배치를 고르시고, 해당 키보드 배치가 여기 없으면 어느정도 비슷한 키보드 배치를 고르십시오. 시스템 설치를 모두 마치면 더 많은 종류의 키보드 배치 중에서 하나를 고를 수 있게 됩니다. (**dpkg-reconfigure keyboard-configuration**를 실행하십시오.)

반전 표시를 해당 키보드로 옮기고 **Enter**를 누르십시오. 화살표 키로 반전 표시를 움직입니다. 화살표 키는 모든 언어의 키보드에 대해 동일하므로, 키보드 설정과는 상관이 없습니다.

6.3.1.4 데비안 설치 프로그램 ISO 이미지 찾기

hd-media에 설치하는 경우, 설치하다가 설치 파일의 나머지를 읽을 때 데비안 설치 프로그램 ISO 이미지를 찾아서 마운트합니다. **iso-scan** 구성 요소가 그 일을 합니다.

[1] 기술적인 용어로 언어에 대해 국가 코드가 다른만큼, 여러 로캘이 존재합니다.

[2] 우선 순위가 중간 혹은 낮은에서는 선택한 언어로 유효한 지역 중 항상 마음에 드는 것을 선택할 수 있습니다(여러 개있는 경우).

[3] 레가시 로캘은 UTF-8 문자 인코딩을 사용하지 않고, ISO 8859-1(서유럽 언어로 사용) 또는 EUC-JP(일어로 사용) 등의 오래전 문자 인코딩을 사용합니다.

처음에 **iso-scan**은 알려진 파일 시스템을 사용하는 블록 장치(파티션과 논리 볼륨 등)를 자동으로 마운트하고, `.iso`(더 말하면 `.ISO`)로 끝나는 파일 이름을 순서대로 검색합니다. 단 처음 시도에서 루트 디렉터리와 하위 디렉터리 밖에 검색하지 않습니다(즉 `/whatever.iso`과 `/data/whatever.iso`을 감지하지만 `/data/tmp/whatever.iso`는 찾지 않습니다 것입니다). ISO 이미지를 찾으면, **iso-scan**는 그 이미지가 올바른 데비안 ISO 이미지인지 아닌지 판단하려고 그 내용을 확인합니다. 전자의 경우는 완료되지만 후자의 경우 **iso-scan**은 다른 이미지를 찾습니다.

앞에서 설치 ISO 이미지를 찾는 데 실패하면, **iso-scan**에서는 계속해서 이미지를 찾을 지 여부를 물어봅니다. 그러면 맨 위의 디렉터리만 찾는 게 아니라, 모든 파일 시스템을 뒤져봅니다.

iso-scan에서 설치 프로그램 ISO 이미지를 찾지 못했다면, 원래 운영 체제로 다시 시작해서 이미지 이름이 올바른 지(`.iso`로 끝나는 지), `debian-installer`가 인식할 수 있는 파일 시스템에 들어 있는지, 파일이 손상되지 않았는지(체크섬 확인) 확인하십시오. 경험 많은 유닉스 사용자라면 다시 시작하지 않고 두번째 콘솔에서 할 수도 있습니다.

ISO 이미지가 들어 있는 파티션은 설치 과정에서 다시 사용할 수 없습니다. 이 파티션을 설치 프로그램에서 사용하기 때문입니다. 이 제한을 피하려면, 메모리가 충분하다는 가정 하에, 설치 프로그램에서 ISO 이미지를 마운트하기 전에 RAM에 복사할 수 있습니다. 이 기능은 우선순위가 낮은 `iso-scan/copy_iso_to_ram` debconf 질문에 따라 결정됩니다. (이 질문은 메모리가 필요한 만큼 많을 경우에만 사용할 수 있습니다.)

6.3.1.5 네트워크 설정하기

이 단계에서 시스템에 네트워크 장치를 두 개 이상 찾으면, 어떤 장치를 주요 네트워크 인터페이스로 사용할지 질문을 받게 됩니다. 주요 네트워크 인터페이스란 설치할 때 사용할 인터페이스를 말합니다. 이 인터페이스 외의 인터페이스는 이 시점에서는 설정하지 않습니다. 설치가 다 끝난 다음에 네트워크 장치를 추가로 설정할 수 있습니다. interfaces(5) 맨 페이지를 참고하십시오.

6.3.1.5.1 네트워크 자동 설정

기본으로 `debian-installer`는 가능하면 자동으로 컴퓨터의 네트워크 설정을 시도합니다. 자동 설정이 실패한 경우는 여러가지 원인이 있습니다. 네트워크 케이블이 빠졌다든지, 자동 설정에 필요한 네트워크 환경이 아니라든지 등 여러가지 문제 때문에 실패할 수 있습니다. 오류를 확인하려면 4번째 콘솔에서 오류 메시지를 보십시오. 어떤 상황이든 다시 시도하거나, 수동으로 설정할 지 여부를 물어봅니다. 가끔 네트워크 서비스가 자동 설정에 필요한 응답이 느릴 수도 있으니, 정상이 분명하다고 확신한다면 자동 설정을 다시 시도해 보십시오. 자동 설정이 계속 실패하면 수동 네트워크 설정으로 들어갈 수 있습니다.

6.3.1.5.2 네트워크 수동 설정

수동 네트워크 설정에서는 네트워크에 관한 여러가지 정보를 차례대로 물어봅니다. `IP 주소`, `네트마스크`, `게이트웨이`, `네임 서버 주소`, `호스트이름`을 물어봅니다. 또 무선 네트워크 인터페이스가 있다면, `무선 ESSID`(무선 네트워크 이름) 및 `WEP 키` 또는 `WPA/WPA2 암호`를 물어봅니다. 3.3절의 답을 채워 넣으십시오.

참고

알아두면 편리할 수도 있고 아닐 수도 있는 기술적인 정보: 이 프로그램에서는 네트워크 IP 주소가 시스템의 IP 주소와 네트마스크를 비트 AND한 값이라고 가정합니다. 브로드캐스트 주소는 시스템의 IP 주소와 네트마스크의 비트 NOT한 값을 OR한 값이라고 가정합니다. 또 게이트웨이도 임의로 추정합니다. 수동 설정에서 무슨 값을 써야 할지 잘 모르겠다면, 시스템의 추정값을 사용해 보십시오. 일단 시스템을 설치한 다음에 필요하다면 `/etc/network/interfaces` 파일을 편집해서 이 설정을 바꿀 수 있습니다.

6.3.1.5.3 IPv4 및 IPv6

데비안 GNU/리눅스 7.0("Wheezy")부터, `debian-installer`에서 IPv6를 "전통적인" IPv4와 마찬가지로 지원합니다. 모든 IPv4와 IPv6 조합을(IPv4 전용, IPv6 전용, 동시 사용) 지원합니다.

IPv4의 자동 설정은 DHCP(Dynamic Host Configuration Protocol)를 이용합니다. IPv6 자동 설정은 NDP(Neighbor Discovery Protocol, 재귀적 DNS 서버 설정(RDNSS) 기능 포함)를 사용한 상태 없는 자동 설정, DHCPv6를 이용한 상태 있는 자동 설정과, 상태 있는/없는 방식을 혼합한 자동 설정도(NDP를 통해 주소 설정, DHCPv6를 통해 추가 파라미터 설정) 지원합니다.

6.3.2 사용자 및 암호 설정

클럭 설정 직전에 설치는 "root" 계정 및 초기 사용자 계정을 설정합니다. 다른 사용자 계정은 설치 후 작성하십시오.

6.3.2.1 루트 암호 설정

루트(root) 계정은 수퍼유저라고도 합니다. 이 계정은 시스템의 모든 보안 장벽을 그냥 통과할 수 있습니다. 루트 계정은 시스템 관리를 할 경우에만 사용해야 하고, 가능한 한 짧은 시간만 사용해야 합니다.

암호를 만들 때 적어도 6자 이상이고, 대문자와 소문자와 특수 문자를 모두 포함해야 좋습니다. 루트 암호를 설정할 때 좀 더 주의를 기울여 주십시오. 루트 계정은 권한이 막강합니다. 사전에 나와있는 단어나 추측할 수 있는 개인 정보는 암호에 사용하지 마십시오.

누가 루트 암호를 말해 달라고 하면 조심하십시오. 한 시스템의 관리자가 여러 명인 경우가 아니라면, 루트 암호는 다른 사람에게 알려주면 안 됩니다.

여기서 "root"에 대한 암호를 지정하지 않은 경우, 루트 계정은 사용할 수 없게 되지만 뒤에 **sudo** 패키지를 설치해서 새 시스템에서 관리 작업을 할 수 있게 됩니다. 기본값으로 시스템에서 맨 처음에 만든 사용자는 **sudo** 명령으로 루트 사용자가 될 수 있게 허용합니다.

6.3.2.2 일반 사용자 만들기

여기에서 일반 사용자 계정을 만들 것인지를 물어봅니다. 이 계정은 주로 사용하는 개인 로그인 계정입니다. 일상적인 용도나 개인 로그인에 루트 계정을 사용하면 안 됩니다.

루트 권한의 사용을 피하는 이유 중 하나는 루트 계정으로는 아주 쉽게 복구하기 힘든 손상을 입힐 수 있기 때문입니다. 다른 이유로는 수퍼유저의 권한을 이용해 숨어서 시스템의 보안을 침해하는 프로그램인, 트로이 목마 프로그램을 실행하도록 속을 수도 있기 때문입니다. 유닉스 시스템 관리에 대한 완만한 책에서는 모두

이 주제에 대해 좀 더 자세하게 설명하고 있습니다. 처음 접한다면 보안 문제에 대한 책을 한 권 정도 읽어 보십시오.

먼저 전체 이름을 입력하고 사용자 계정으로 사용할 이름을 입력합니다. 사용자 계정은 이름같은 걸 사용하면 충분하고, 실제로 이름이 기본 값입니다. 마지막으로 이 계정의 암호를 입력하십시오.

설치가 끝난 다음에 언제든지 또 계정을 만드려면, **adduser** 명령을 사용하십시오.

6.3.3 시계 및 시간 설정

설치 프로그램은 시스템 시각을 정확히 맞추려고 먼저 인터넷의 타임 서버에 (NTP 프로토콜 사용) 연결합니다. 연결이 실패하면, 부팅할 때 시스템 시계에서 읽은 시각과 날짜가 올바르다고 가정합니다. 설치 과정에서 시스템 시각을 수동으로 맞추는 방법은 없습니다.

설치 과정에서 처음으로 선택한 지역에 따라 해당 위치에 해당하는 시간대의 목록을 표시합니다. 여러분의 위치에 시간대가 하나 밖에 없고 기본 설치를 수행하는 경우, 시간대 목록을 나열하지 않고, 그 하나의 시간대에 있다고 가정합니다.

전문가 모드 및 우선 순위에서 설치하는 경우 표준 시간대 "협정세계시"(UTC)를 사용한다는 옵션이 추가됩니다.

무슨 이유에서든 선택한 위치에 해당하지 않는 시간대를 이용하려면, 두 가지 방법이 있습니다.

1. 간단한 방법은 설치를 완료하고 새로 부팅한 후 다른 시간대를 선택하는 것입니다. 다음과 같은 명령을 사용합니다:

   ```
   # dpkg-reconfigure tzdata
   ```

2. 다른 방법으로 설치 시스템이 부팅할 때 **time/zone=값** 파라미터를 넘겨서 시간대를 설정할 수도 있습니다. 이 값은 올바른 시간대 값이어야 합니다. 예를 들어 **Europe/London**이나 **UTC**가 있습니다.

자동 설치에 preseed를 사용하여 시간대를 원하는 값으로 설정할 수 있습니다.

6.3.4 파티션하기 및 마운트 위치 선택

여기에서 마지막으로 하드웨어 검색을 하면, debian-installer는 필요한 기능을 모든 갖추고, 사용자의 필요에 맞게 진짜 설치 작업을 할 준비를 갖추게 됩니다. 제목이 말하듯 다음 몇 개의 컴포넌트는 디스크를 파티션하고, 파일 시스템을 만들고, 마운트 위치를 지정하고, 또 필요하다면 LVM, RAID, 암호화 장치와 같은 관련 설정을 합니다.

파티션하는 게 불안하거나 자세히 알고 싶으시면, 부록 C 부분을 참고하십시오.

먼저 선택에 따라, 전체 드라이브나 드라이브의 빈 공간을 자동으로 파티션할 수 있습니다. 이 방법을 "자동" 파티션이라고 합니다. 자동 파티션을 하지 않으려면, 메뉴에서 수동으로를 선택하십시오.

6.3.4.1 지원하는 파티션 옵션

debian-installer에서 사용하는 파티션 도구는 꽤 만능입니다. 그러면 다양한 파티션 테이블, 파일 시스템 코드 블록 장치를 사용하여 많은 다른 파티션 구성표를 만들 수 있습니다.

정확히 어떤 옵션을 사용할 수 있는지는 주로 아키텍처에 따라 다르지만, 그 다른 요인도 있습니다. 예를 들어, 내부 메모리가 제한된 시스템에서는 몇 가지 옵션을 사용할 수 없습니다. 또한 기본도 변할지도 모릅니다. 예를 들어, 대용량 하드 디스크에 대한 기본 파티션 테이블의 유형은 더 작은 하드 디스크의 것과

다른 경우가 있습니다. debconf 우선 순위가 중 또는 낮은 설치를 하는 경우에만 몇 가지 옵션을 변경할 수 있습니다. 더 높은 우선 순위의 경우는 실제적인 값이 기본적으로 사용됩니다.

설치 프로그램은 다양한 형태의 고급 파티션 및 저장 장치를(대부분의 경우 함께) 지원합니다.

- 논리 볼륨 관리자(LVM)
- 소프트웨어 RAID

  지원하는 RAID 레벨은 0, 1, 4, 5, 6, 10 입니다.
- 암호화
- Multipath (실험적)

  자세한 정보는 Wiki를 참조하십시오. 현재 다중 경로는 설치 시작시 활성화된 경우에만 사용할 수 있습니다.

다음 파일 시스템을 지원합니다.

- ext2r0, ext2, ext3, ext4

  대부분의 경우 기본 파일 시스템 ext4가 선택되어 있습니다. 파티션을 사용할 때 `/boot` 파티션의 기본은 ext2가 선택됩니다.
- jfs (모든 아키텍처에서 사용하지 못할 수도 있습니다)
- xfs (모든 아키텍처에서 사용하지 못할 수도 있습니다)
- reiserfs (옵션; 모든 아키텍처에서 사용할 수 있는 것은 없습니다)

  Reiser 파일 시스템은 더 이상 기본적으로 지원되지 않습니다. 설치 프로그램이 중 또는 낮은 debconf 우선 순위로 실행 시키면 `partman-reiserfs` 구성 요소를 선택하여 사용할 수 있습니다. 버전 3에서만 지원합니다.
- jffs2

  플래시 메모리를 읽을 때 일부 시스템에서 사용합니다. 새로운 jffs2 파티션을 만들 수 없습니다.
- FAT16, FAT32

6.3.4.2 자동 파티션하기

자동 파티션하는 경우, 세가지 방법이 있습니다: 하드 디스크에서 직접 파티션을 만들거나(전통적인 방법), 논리 볼륨 관리를(LVM) 사용하거나, 암호화한 LVM을[4] 사용하는 것입니다.

> 참고
>
> 
>
> 아키텍처에 따라(암호화한) LVM을 사용하는 옵션을 사용하지 못할 수도 있습니다.

LVM이나 암호화한 LVM을 사용하는 경우, 설치 프로그램은 대부분의 파티션을 한 개의 큰 파티션 안에 만듭니다. 이 방법의 장점으로 이 큰 파티션 안의 파티션은 나중에 비교적 쉽게 크기를 바꿀 수 있습니다.

[4] 설치 프로그램에서 LVM 볼륨 그룹을 256비트 AES 키를 이용해 암호화하고 커널의 "dm-crypt" 기능을 사용합니다.

암호화한 LVM의 경우 특수한 암호를 알지 못하면 이 큰 파티션을 읽지 못하므로, (개인적인) 데이터에 대해 더 보안에 안전합니다.

암호화된 LVM을 사용할 때, 설치 프로그램은 디스크에 임의의 데이터를 써 넣어서 디스크를 지웁니다. 이렇게 하면 보안을 더욱 높이겠지만(디스크의 어느 부분을 사용하고 있는지 추적하는 게 불가능하고, 예전에 설치했던 데이터를 지우기 때문입니다), 디스크 크기에 따라 시간이 오래 걸릴 수도 있습니다.

참고

LVM이나 암호화한 LVM을 사용해 자동 파티션을 하는 경우, LVM을 설정하는 동안 파티션 테이블의 일부를 바꿔야 합니다. 이렇게 하면 해당 하드 디스크에 있는 데이터가 전부 지워지고 되돌릴 수 없습니다. 설치 프로그램이 파티션 테이블을 디스크에 쓰기 전에 계속해도 좋을 지 확인 질문을 합니다.

전체 디스크에 대해 파티션을 선택한 경우 (일반적인 파티션이나 LVM(또는 암호화한 LVM)에서), 먼저 선택한 디스크를 사용해도 되는지 묻습니다. 여러 디스크가 있는 경우 모든 디스크가 나열되어 올바르게 선택되어 있는지 확인하십시오. 순서는 평상시 사용하고 있는 것과 다를 수 있습니다. 디스크 크기를 확인할 수 있습니다. 전체 디스크에 대해 자동 파티션 하기를 (일반적인 파티션이나 LVM(또는 암호화한 LVM)에서) 선택했다면, 사용하려는 디스크를 선택하라는 질문을 맨 처음에 받게 됩니다. 디스크가 모두 목록에 있는 지 확인하십시오. 디스크가 여러 개 있을 경우 반드시 올바른 디스크를 선택하십시오. 목록에 나오는 순서는 경우에 따라 달라질 수 있습니다. 디스크 용량으로도 어떤 디스크인지 확인할 수 있습니다.

여기서 결국, 디스크의 모든 데이터가 손실되었지만 디스크를 쓰기 전에 계속해도 좋을지 항상 질문을 확인합니다. 일반적인 파티션 방법을 선택하면 종료하기 전에 취소할 수 있습니다. 반면 LVM(또는 암호화한 LVM)을 사용하는 경우 취소할 수 없습니다. 선택한 디스크에 있는 모든 데이터를 영영 잃어버리게 될 것입니다. 하지만 디스크에 바꾼 사항을 쓰기 전에 언제나 확인 질문을 합니다. 일반적인 파티션 방법을 사용한다면 끝날 때 바꾼 사항을 취소할 수 있습니다. 반면 LVM(또는 암호화한 LVM)을 사용하는 경우에는 취소가 불가능합니다.

그 다음에 아래의 표에 나온 방식 중 하나를 선택할 수 있습니다. 이 방식은 각각 장단점이 있고, 부록 C에서 설명합니다. 잘 모르겠으면 첫번째를 선택하십시오. 명심해 둬야 할 점으로, 자동 파티션할 때는 어느정도 최소한의 빈 공간이 필요합니다. 최소 1GB의 공간이 없으면(방식에 따라 이 최소 용량은 다릅니다) 자동 파티션은 실패합니다.

파티션 방식	**최소 공간**	**만들 파티션**
모두 한 파티션에 설치	600MB	/, 스왑
/home 파티션 분리	500MB	/, /home, 스왑
/home, /var, /tmp 파티션 분리	1GB	/, /home, /var, /tmp, 스왑

LVM(또는 암호화한 LVM)을 사용해 자동 파티션하는 경우, 설치 프로그램은 별도의 /boot 파티션을 만듭니다. 그 외의 파티션(스왑 파티션 포함)은 LVM 파티션 안에 만듭니다.

파티션 방법을 선택하면, 그 다음 화면에서 새로 만든 파티션 테이블이 나타납니다. 여기에는 파티션을 포맷할 형식과 마운트할 위치에 대한 정보도 들어 있습니다.

파티션 목록은 다음과 같이 나타납니다:

```
SCSI1 (0,0,0) (sda) - 6.4 GB WDC AC36400L
      #1 primary   16.4 MB  B f ext2       /boot
```

```
       #2 primary  551.0 MB       swap        swap
       #3 primary    5.8 GB       ntfs
          pri/log    8.2 MB       FREE SPACE

  SCSI2 (1,0,0) (sdb) - 80.0 GB ST380021A
       #1 primary   15.9 MB       ext3
       #2 primary  996.0 MB       fat16
       #3 primary    3.9 GB       xfs         /home
       #5 logical    6.0 GB    f  ext4        /
       #6 logical    1.0 GB    f  ext3        /var
       #7 logical  498.8 MB       ext3
```

위의 예에서는 하드드라이브가 2개이고 여러 개 파티션으로 나눠져 있습니다. 첫번째 디스크에는 빈 공간이 있습니다. 각 파티션 줄에는 파티션 번호, 종류, 크기, 추가 플래그, 파일시스템, 그리고 마운트 위치를(마운트 위치가 따로 있는 경우) 표시합니다. 주의: 위와 같은 파티션은 자동 파티션하면 만들 수 없고, 수동으로 만들면 이렇게 될 수도 있다는 걸 안내하는 것 뿐입니다.

여기까지가 자동 파티션입니다. 자동으로 만든 파티션 테이블이 마음에 들면, 메뉴에서 파티션 나누기를 마치고 바뀐 사항을 디스크에 쓰기를 선택해서 새 파티션 테이블을 실제로 적용합니다(이 절의 맨 뒤 부분에서 설명합니다). 마음에 들지 않으면, 파티션에 바뀐 사항을 취소를 선택해서 자동 파티션을 다시 실행하거나, 자동으로 만들어 준 파티션을 아래에서 설명하는 것처럼 수동으로 바꿀 수도 있습니다.

6.3.4.3 수동 파티션하기

수동 파티션을 선택하면 기존 파티션 테이블을 마운트 위치없이 표시되는 것을 제외하고 위와 같은 화면이 표시됩니다. 파티션 테이블을 수동으로 만드는 방법과 새로운 데비안 시스템 파티션의 사용법에 대해서는 이 절의 나머지 부분에서 설명합니다.

파티션도 없고 빈 공간도 없는 새 디스크라면, 새로운 파티션 테이블을 만들지 여부를 물어봅니다. (그래야 새 파티션을 만들 수 있습니다.) 그 다음에 "빈 공간"이라는 줄이 해당 디스크 이름 아래에 나타납니다.

빈 공간을 선택하면 새 파티션을 만들 수 있습니다. 크기와 종류(주 파티션 아니면 논리 파티션), 위치와(빈 공간에서 처음 아니면 끝) 같은 일련의 간단한 질문에 응답해야 합니다. 그러면 새 파티션에 대한 자세한 정보를 얻을 수 있습니다. 주요 설정은 파일 시스템 파티션에 있는 경우 스왑 , 소프트웨어 RAID, LVM, 암호화한 파일 시스템 으로 사용하거나 전혀 사용 여부를 결정 하는 이용 방법:입니다. 기타 설정은 마운트 포인트와 마운트 옵션, 부팅 가능 플래그 같은 파티션 용도에 따라 설정이 있습니다. 미리 선택된 기본값이 마음에 들지 않으면 자유롭게 원하는 것으로 변경하십시오. 예를 들어, 옵션 이용 방법:를 선택하면 스왑, 소프트웨어 RAID, LVM, 또는 사용하지 않기를 선택할 수 있습니다. 새 파티션이 마음에 들면, 파티션 준비를 마쳤습니다를 선택하여 **partman**의 메인 화면으로 돌아갑니다.

파티션에서 뭔가 바꾸려고 한다면, 해당 파티션을 선택하십시오. 그러면 파티션 설정 메뉴가 나타납니다. 새 파티션을 만들 때와 같은 화면이기 때문에, 여기서도 마찬가지의 같은 옵션을 설정합니다. 처음 보면 잘 이해가 되지 않을 수도 있는 부분이 있는데, 파티션의 크기 항목을 이용해서 파티션의 크기를 바꿀 수 있다는 점입니다. 이 기능이 동작하는 파티션은 fat16, fat32, ext2, ext3 및 스왑입니다. 이 메뉴에서 파티션을 지워 버릴 수도 있습니다.

최소한 파티션을 두 개 만들도록 하십시오. 한 개는 루트 파일 시스템이고 (/에 마운트합니다), 다른 하나는 스왑입니다. 루트 파일 시스템을 마운트하지 않으면, 그 문제를 바로잡기 전에는 다음으로 진행하지 없습니다.

partman의 기능은 설치 프로그램 모듈에 따라 확장되지만, 시스템의 아키텍처에 따라 다릅니다. 모든 기

능을 볼 수 없다면, 필요한 모듈을 모두 읽어들였는지 확인하십시오. (예를 들어 partman-ext3, partman-xfs, 아니면 partman-lvm)

파티션이 마음이 들면, 파티션 메뉴에서 파티션 나누기를 마치고 바뀐 사항을 디스크에 쓰기를 선택하십시오. 디스크에 바뀐 점에 대한 요약이 나타나고 이대로 파일 시스템을 만들지 확인합니다.

6.3.4.4 멀티디스크 장치 설정하기(소프트웨어 RAID)

컴퓨터에 하드 드라이브가 여러 개 있는 경우[5], **partman-md** 명령으로 드라이브의 성능을 향상시키거나, 데이터 안정성을 높일 수 있습니다. 이렇게 하는 걸 멀티디스크 장치라고 합니다. (더 자주 쓰이는 다른 말로 RAID라고 합니다.)

기본적으로 멀티디스크는 여러 디스크에 있는 여러 개의 파티션으로 하나의 논리 장치를 만드는 걸 말합니다. 그러면 이 논리 장치는 일반 파티션처럼 사용할 수 있습니다. (예를 들어 **partman**에서 포맷하고 마운트 위치를 지정하고 따위를 할 수 있습니다.)

어떤 종류의 멀티디스크 장치를 만드냐에 따라 어떤 이점이 있는지 달라집니다. 현재 지원하는 종류는:

RAID0 주로 성능 향상이 목적입니다. RAID0는 들어오는 데이터를 스트라이프(stripe)로 나눠서 배열의 각 디스크에 똑같이 분산시킵니다. 이렇게 하면 읽기/쓰기 작업의 속도가 빨라집니다. 하지만 디스크중에 하나라도 망가지면, 모든 데이터를 잃게 됩니다. (망가지지 않은 디스크에 일부 데이터가 남아 있겠지만, 또 다른 부분이 망가진 디스크에 있었기 때문입니다.)

RAID0은 비디오를 편집하는 파티션에 많이 사용합니다.

RAID1 안정성을 최우선으로 할 때 적합합니다. RAID1은 여러 개의(보통 두개) 같은 크기의 파티션으로 구성되어 있고, 여기서 각 파티션은 정확히 같은 데이터를 담고 있습니다. 이게 근본적으로 3가지 의미가 있습니다. 첫째로 하나의 디스크가 망가지더라도, 나머지 디스에 데이터가 미러되어 있습니다. 둘째로 디스크의 전체 용량보다 작은 용량만 사용할 수 있습니다. (좀더 정확히 말해, RAID에서 가장 작은 파티션입니다.) 세번째로 파일 읽기는 로드 밸런싱으로 서버의 성능을 향상시켜, 파일서버와 같이 디스크 읽기가 쓰기보다 많은 경우 부담이 줄어듭니다.

RAID5 속도, 안정성, 데이터 중첩을 적당히 조화시킨 것입니다. RAID5는 들어오는 모든 데이터를 스트라이프로 나누고 각각을 하나의 디스크가 아니라(즉 RAID0와는 달리) 모두에게 분배합니다. RAID0와는 다르게 RAID5는 디스크에 쓸 정보의 패리티 정보를 계산합니다. 패리티 디스크는 고정되어 있지 않고(고정되어 있으면 RAID4라고 합니다) 정기적으로 바뀝니다. 디스크중에 하나가 망가지면, 없어진 부분을 나머지 데이터와 패리티를 이용해서 계산해 냅니다. RAID5는 최소한 3개의 파티션이 있어야 합니다. 배열에 디스크를 하나 더 사용해서 망가진 디스크를 대체하도록 만들 수도 있습니다.

이렇게 RAID5는 RAID1과 비슷한 정도의 안정성을 가지면서, 데이터를 덜 중복합니다. 한편 패리티 계산때문에 RAID0보다는 쓰는 속도가 느립니다.

RAID6 RAID5와 비슷하지만 패리티 장치를 하나가 아니라 두 개를 사용합니다.

RAID6 어레이는 디스크 실패가 두 번 일어나도 살아남을 수 있습니다.

RAID10 입력 데이터를 n개 복사물로 만든 다음 파티션에 분배해서 같은 데이터가 같은 장치에 저장되지 않도록 합니다. n의 기본값은 2이지만 전문가 모드에서는 다른 값으로 설정할 수 있습니다. 사용하는 파티션 개수는 최소한 n개입니다. RAID10은 복사물을 분배하는 레이아웃 방식이 여러가지가 있습니

[5]물론 한 개의 물리 드라이브에 있는 여러개의 파티션에서 멀티디스크 장치를 만들 수도 있지만, 그렇게 해 봤자 좋은 점이 전혀 없습니다.

다. 기본 레이아웃은 니어 카피(Near copies)입니다. 니어 카피에서는 모든 복사물의 디스크 오프셋이 같습니다. 파 카피(Far copies)에서는 복사물의 오프셋이 다릅니다. 오프셋 카피(Offset copies)는 개개 복사물이 아니라 전체 스트라이프를 반복합니다.

요약하면:

종류	**장치 최소 개수**	**예비 장치**	**디스크가 망가져도 버티는지?**	**사용 가능 공간**
RAID0	2	아니오	아니오	RAID에서 가장 작은 파티션의 크기 x 장치 개수
RAID1	2	옵션	예	RAID에서 가장 작은 파티션의 크기
RAID5	3	옵션	예	가장 작은 파티션의 크기 x (RAID의 장치 개수 - 1)
RAID6	4	옵션	예	가장 작은 파티션의 크기 x (RAID의 장치 개수 - 2)
RAID10	2	옵션	예	전체 파티션 나누기 단위 복사물의 개수 (기본값 2)

소프트웨어 RAID를 좀 더 알고 싶으시면, Software RAID HOWTO를 읽어 보십시오.

멀티디스크 장치를 만드려면, 구성할 파티션을 RAID에 사용한다고 표시해야 합니다. (파티션 설정 메뉴의 **partman**에서 용도: → RAID의 물리 볼륨을 선택하십시오.

참고

사용하려고 하는 파티션 방식에서 시스템이 부팅할 수 있도록 하십시오. 루트 파일 시스템으로 RAID를 사용하는 경우 `/boot`에 대한 파일 시스템을 따로 만드는 게 보통입니다. 보통 부트로는 RAID1의 미러링을 지원합니다. (스트리핑은 지원하지 않습니다!) 그러므로 `/`에는 RAID5를 사용하고 `/boot`에 RAID1을 사용하는 것도 한 가지 방법입니다.

그 다음 **partman** 메뉴에서 소프트웨어 RAID 설정을 선택합니다. (최소한 한 파티션을 RAID의 물리 볼륨으로 표시해야만 메뉴가 나타납니다.) **partman-md**의 첫번째 화면에서 멀티디스크 장치 만들기를 선택하기만 하면 지원하는 멀티디스크 장치의 종류 목록이 나옵니다. 거기에서 하나를(예를 들어 RAID1) 고르십시오. 그 다음은 여기서 어떤 종류의 멀티디스크를 선택했냐에 따라 달라집니다.

- RAID0는 간단합니다. RAID 파티션의 목록이 나타나고 거기에서 멀티디스크를 구성할 파티션을 선택하기만 하면 됩니다.
- RAID1은 약간 더 까다롭습니다. 먼저 멀티디스크를 구성할 활성 장치의 개수 및 예비 장치의 개수를 입력합니다. 그리고 RAID 파티션 목록에서 무엇을 활성 파티션과 예비 파티션으로 할 지 결정합니다. 여기서 선택한 파티션 개수는 앞에서 입력한 개수와 일치해야 합니다. 걱정할 필요는 없습니다. 실수로 파티션 개수가 틀렸다고 해도, 개수가 맞아야 `debian-installer`가 다음으로 진행합니다.
- RAID5는 RAID1과 비슷한 설정을 하지만, 최소한 3개의 활성 파티션을 사용해야 한다는 점이 다릅니다.
- RAID6는 RAID1과 비슷한 설정을 하지만, 최소한 4개의 활성 파티션을 사용해야 한다는 점이 다릅니다.
- RAID10도 RAID1과 비슷한 설정을 하지만 전문가 모드에서는 다릅니다. 전문가 모드에서는 `debian-installer`에서 레이아웃을 물어봅니다. 레이아웃은 두 부분이 있습니다. 첫 번째는 레이아웃 종류입니다. 레이아웃 종류는 `n` (니어 카피, near copies), `f` (파 카피, far copies), `o` (오프셋 카피, offset copies) 중의 하나입니다. 두 번째 부분은 데이터의 복사물 개수입니다. 이 숫자는 최소한 활성 장치의 개수만큼이어야 합니다. 그래야 모든 카피가 다른 디스크에 분배됩니다.

여러가지 종류의 멀티디스크를 동시에 사용하는 것도 물론 가능합니다. 예를 들어 3개의 200 GB 하드드라이브를 멀티디스크에 사용할 때, 각 디스크에 100 GB 파티션이 두개씩 있다고 할 때, 각 3개 디스크의 첫번째 파티션을 RAID0로 묶고(빠른 300GB 비디오 편집 파티션) 나머지 3개 파티션을(2개 활성, 1개 예비) RAID1으로(`/home`에 사용할 안정성 높은 100GB 파티션) 사용할 수 있습니다.

원하는 대로 멀티디스크 장치를 만든 다음에, **partman-md**에서 마치기를 선택하고 **partman**으로 돌아가 새로 만든 멀티디스크 장치에 파티션을 만들고 마찬가지로 마운트 위치와 같은 속성을 부여할 수 있습니다.

6.3.4.5 LVM (논리 볼륨 관리자) 설정하기

시스템 관리자나 "고급" 사용자 입장에서 컴퓨터를 사용한다면, 어떤 디스크 파티션이(보통 가장 중요한 파티션) 공간이 부족하고 다른 파티션은 공간이 남아서 데이터를 옮기고 심볼릭 링크를 걸고 하는 등의 작업으로 상황을 해결해야 했던 경험이 있을 겁니다.

이러한 상황을 피하려면, LVM(Logical Volume Manager, 논리 볼륨 관리자)을 사용할 수 있습니다. 간단히 말해 LVM을 사용하면 여러 파티션을(물리 볼륨) 하나의 가상 디스크로(볼륨 그룹) 합칠 수 있고, 그걸 다시 가상 파티션으로(논리 볼륨) 나눌 수 있습니다. 중요한 점은 논리 볼륨이(그리고 그 밑에 있는 볼륨 그룹이) 여러 개의 물리 파티션에 걸쳐 있을 수 있다는 점입니다.

기존의 160GB `/home` 파티션에 용량이 더 필요하다면, 300GB 디스크를 컴퓨터에 붙이고, 이 디스크를 현재 볼륨 그룹에 포함시키고, `/home`이 들어 있는 논리 볼륨 크기를 늘리면 됩니다. 그러면 파티션은 460GB가 되어 남는 공간이 더 생깁니다. 물론 이 예는 너무 간단하게 설명한 것입니다. 아직 읽지 않으셨다면 LVM HOWTO를 참고하십시오.

-"`debian-installer`의 LVM 설정은 아주 간단하고 **partman** 안에서 지원합니다. 먼저 LVM의 물리 볼륨으로 사용할 파티션을 표시합니다. 이 작업은 파티션 설정 메뉴에서 용도: LVM의 물리 볼륨을 선택합니다.

주의

주의하십시오. 새로운 LVM 방식으로 설정하면 LVM 타입 코드로 표시된 모든 파티션의 모든 데이터가 삭제됩니다. 즉 이미 디스크에 LVM을 사용하고 있고, 그 컴퓨터에 추가로 데비안을 설치하려는 경우, 기존의 LVM 설정이 모두 지워집니다! 파티션의 경우에도 마찬가지입니다. 파티션이 (어떤 이유에서이든) LVM 타입 코드로 잘못 표시되어 있으면서, 실제 내용이 다른 경우 (예를 들어 암호화된 볼륨) 그 내용도 삭제됩니다. 시스템에서 그러한 디스크를 먼저 제거한 다음 새로 LVM 설정을 해야 합니다.

partman 주화면으로 돌아간 다음, LVM(논리 볼륨 관리자) 설정이라는 옵션이 새로 나타납니다. 이 옵션을 선택하면, 파티션 테이블에 바꾼 사항을(있으면) 확인하고, 그 다음에 LVM 설정 메뉴가 나타납니다. 그 메뉴 위에 LVM 설정의 요약이 나타납니다. 메뉴는 상황에 따라 사용할 수 있는 동작만 표시합니다. 가능한 동작은:

- 설정 내역 보기: LVM 장치 구조, 이름, 논리 볼륨의 크기 등을 표시합니다
- 볼륨 그룹 만들기
- 논리 볼륨 만들기
- 볼륨 그룹 삭제
- 논리 볼륨 삭제
- 볼륨 그룹 늘이기
- 볼륨 그룹 줄이기
- 마치기: **partman** 주 화면으로 돌아갑니다

메뉴에서 이 옵션을 이용해 볼륨 그룹을 만들고 그 안에 논리 볼륨을 만드십시오.

partman 주 화면으로 돌아간 다음, 보통 파티션과 마찬가지로 방금 만든 논리 볼륨이 나타납니다. (또 보통 파티션과 마찬가지 방법으로 이용하면 됩니다.)

6.3.4.6 암호화 볼륨 설정하기

debian-installer에서 암호화 파티션을 설정할 수 있습니다. 암호화된 파티션에 파일을 쓰면 즉시 암호화된 형태로 장치에 저장됩니다. 암호화된 데이터에 접근하려면 파티션을 처음에 만들 때 사용한 암호를 입력해야 합니다. 이 기능은 노트북이나 하드 드라이브를 도난 당했을 때 비밀 데이터를 보호하는 목적으로 사용합니다. 훔친 사람은 하드 드라이브에 물리적으로는 접근할 수 있지만, 올바른 암호를 모르면 하드 드라이브의 데이터는 임의의 문자로 보입니다.

암호화해야 할 가장 중요한 두 파티션은 데이터가 들어 있는 홈 파티션과, 동작중에 비밀 데이터가 저장될 수도 있는 스왑 파티션입니다. 물론, 그 외에 하고 싶은 파티션을 암호화할 수 있습니다. 예를 들어 메일 서버나 인쇄 서버가 데이터를 저장하는 /var 라던지, 여러가지 임시 파일을 저장해 둘 수도 있는 /tmp 를 암호화할 수 있습니다. 어떤 사람은 전체 시스템을 암호화하기도 합니다. 어떤 경우에도 암호화하지 말아야 하는 한 가지 예외는 /boot 파티션으로, 암호화한 파티션에서 커널을 읽어들이는 건 현재 불가능합니다. (이제 최신 버전의 GRUB에는 이런 기능이 있지만, 현재 debian-installer에 암호화한 /boot 파티션 지원이 없습니다. 그래서 설정 방법은 별도 문서에 있습니다.)

참고

암호화한 파티션은 암호화하지 않은 경우보다 성능이 떨어집니다. 매번 읽고 쓸 때마다 암호화하고 암호화를 해제해야 하기 때문입니다. 성능 저하 정도는 CPU 속도, 암호화 방법 및 키 길이에 달려 있습니다.

암호화를 사용하려면, 파티션 메뉴에서 빈 공간을 선택해서 새 파티션을 만들어야 합니다. 다른 방법은 기존 파티션을(예를 들어, 일반 파티션이나 LVM 논리 볼륨이나 RAID 볼륨) 선택하는 것입니다. 파티션 설정에서, 용도: 옵션에서 암호화할 물리 볼륨 메뉴를 선택해야 합니다. 그러면 메뉴가 해당 파티션에 대한 몇 가지 암호화 옵션이 나오도록 바뀝니다.

`debian-installer`에서 지원하는 암호화 방식은 dm-crypt(최근 리눅스 커널에 포함, LVM 물리 볼륨 사용 가능)입니다.

암호화 방법으로 **디바이스 매퍼(dm-crypt)**를 선택했을 때 옵션을 봅니다. 마찬가지로 무언가 의심스러우면 기본값을 사용하십시오. 이미 보안을 염두에 두고 기본값이 들어 있습니다.

암호화: aes 이 옵션은 암호화 알고리즘을(싸이퍼(cipher)) 선택합니다. 이 알고리즘으로 파티션의 데이터를 암호화하는 데 사용합니다. `debian-installer`는 현재 다음과 같은 블록 싸이퍼를 지원합니다: aes, blowfish, serpent, 그리고 twofish입니다. 각 알고리즘이 얼마나 좋은지 설명하는 건 이 문서의 범위를 벗어난 얘기지만, 어느 것을 결정할 지 도움이 되는 조언을 하면, 2000년에 미국 표준 기술 연구소에서 21세기에 비밀 정보를 보호할 알고리즘으로 AES를 선택했습니다.

키 크기: 256 여기서는 암호화 키의 길이를 지정합니다. 키의 길이가 크면, 보통 그 암호화의 강력함이 더 증가합니다. 반면에 키 길이를 늘리면, 성능이 떨어집니다. 사용할 수 있는 키의 크기는 싸이퍼에 따라 다릅니다.

IV 알고리즘: xts-plain64 초기화 벡터(Initialization Vector) 혹은 IV 알고리즘은 암호화 기법에서 같은 키의 일반 텍스트(clear text)가 항상 유일한 암호화된 텍스트(cipher text)를 만들어 내도록 합니다. 이렇게 하면 공격하는 사람이 암호화된 데이터에 반복된 패턴을 통해 정보를 알아내지 못하게 됩니다.

사용할 수 있는 여러 가지 방법 중에서, 기본값인 **xts-plain64**이 알려진 공격 방법에서 현재 가장 위험이 적은 방법입니다. 다른 방법은 예전에 설치한(새 알고리즘을 사용할 수 없는) 시스템과의 호환성을 유지할 경우에만 사용하십시오.

암호화 키: 암호 여기서 이 파티션의 암호화 키 종류를 선택합니다.

암호 나중에 입력하게 되는 암호를 기준으로 암호화 키를 계산합니다[6].

무작위 키 암호화한 파티션을 불러올 때마다 임의의 데이터를 만들어서 새 암호화 키를 만듭니다. 다시 말해서, 컴퓨터를 끌 때마다 키가 메모리에서 지워지면서 파티션에 들어 있는 데이터를 잃어버립니다. (물론 가능한 키를 모두 입력해서 추측해 낼 수 있지만, 해당 싸이퍼 알고리즘에 무언가 알려지지 않은 약점이 있지 않는한 그렇게 무작정 추측하는 건 평생 해도 다 못 합니다.)

무작위 키는 스왑 파티션에 주로 사용합니다. 암호를 기억할 필요도 없고 컴퓨터를 끌 때 스왑 파티션에 들어 있는 비밀 정보를 지울 필요가 없기 때문입니다. 하지만, 이렇게 하면 최근의 리눅스 커널에 들어 있는 "디스크에 저장하는 절전(하이버네이션)" 기능을 사용하지 못하게 됩니다. 나중에 부팅할 때 스왑 파티션에 저장된 데이터를 복구할 방법이 없기 때문입니다.

[6] 암호를 키로 사용한다는 건 현재는 파티션을 LUKS로 설정한다는 뜻입니다.

데이터 지우기: 예 파티션에 암호화를 설정하기 전에 임의의 데이터로 채워 넣을 지 결정합니다. 이렇게 하지 않으면 공격자가 파티션의 어느 부분을 사용하고 있고, 어느 부분을 사용하지 않는지 알아챌 수 있기 때문에 이렇게 하기를 권장합니다. 또 예전에 설치한 데이터를 복구하기 어렵게 만듭니다[7].

암호화 파티션에 사용할 파라미터를 선택했으면, 주 파티션 메뉴로 돌아갑니다. 이제 암호화 볼륨 설정이라는 새 메뉴 항목이 생깁니다. 이 항목을 선택하면, 데이터 지우기로 표시한 파티션을 지울 지 및 기타 동작에 대한(파티션 테이블을 쓰기 등) 확인 질문을 합니다. 파티션의 크기가 크면 시간이 좀 걸릴 수 있습니다.

그 다음에 암호를 사용한다고 표시한 파티션에 대해 암호를 입력합니다. 좋은 암호는 8글자보다 길고, 영문자와 숫자와 기타 문자가 섞여 있고, 사전에 들어 있는 일반적인 단어가 들어 있으면 안 되고, 본인의 개인 정보에서 쉽게 유추할 수 있으면(생일, 취미, 애완동물 이름, 가족이나 친척 이름 등) 안 됩니다.

주의

암호를 입력하기 전에, 키보드를 올바르게 설정해서 원하는 문자가 입력되도록 해야 합니다. 잘 모르겠으면, 두번째 가상 콘솔로 바꿔서 프롬프트에서 미리 글자를 타이프해 보면 알 수 있습니다. 그래야 나중에 설치할 때 AZERTY 키보드 배치로 입력했던 암호가 QWERTY 키보드 배치에서 맞지 않는다든지 하는 일이 없습니다. 이런 상황은 여러가지 원인때문에 일어날 수 있습니다. 설치할 때 키보드 배치를 바꾸었거나, 루트 파일 시스템의 암호를 입력할 때 아직 예전에 사용했던 키보드 배치를 설정하지 못한 상태일 경우에 이런 일이 일어날 수 있습니다.

암호화 키를 만드는 데 암호 외의 방법을 사용한다면, 그 암호화 키를 이제 만들게 됩니다. 현재 설치 상태에서는 충분한 양의 엔트로피를 얻지 못했을 수도 있기 때문에, 키를 만드는 데 오래 걸릴 수 있습니다. 엔트로피를 만들어 내면 이 과정을 좀 더 빠르게 할 수 있습니다. 예를 들어서 키를 마구 누른다든지, 두번째 가상 콘솔로 가서 쉘로 바꾼 다음에 네트워크와 디스크를 사용한다던지(파일 다운로드, 큰 파일을 /dev/null로 보낸다든지 등) 하면 됩니다. 이 과정을 암호화할 모든 파티션에 대해서 반복합니다.

주 파티션 메뉴로 돌아오면, 암호화한 파티션이 일반 파티션과 똑같이 설정할 수 있게 보입니다. 다음의 예제는 dm-crypt로 암호화한 볼륨입니다.

```
암호화한 볼륨(sda2_crypt) - 115.1 GB 리눅스 device-mapper
     #1 115.1 GB  F ext3
```

이제 해당 볼륨을 가리킬 마운트 위치를 지정하고 (기본값이 마음에 들지 않으면) 파일 시스템 종류를 설정하십시오.

주의해야 할 사항이 있습니다. 괄호 안에 들어 있는 ID와(이 경우 sda_crypt0) 각 암호화 볼륨에 지정한 마운트 위치입니다. 나중에 새 시스템으로 부팅할 때 이 정보가 필요합니다. 일반 부팅 과정과 암호화를 사용하는 부팅 과정의 다른 점은 뒤의 7.2절에서 설명합니다.

파티션 방법이 마음에 들면, 설치를 계속하십시오.

6.3.5 베이스 시스템 설치하기

이 단계는 문제가 발생할 가능성이 거의 없지만, 설치할 때 베이스 시스템 전체를 다운로드하고, 확인하고, 압축을 풀기 때문에 가장 많은 시간을 소모하는 단계이기도 합니다. 컴퓨터가 느리거나 네트워크 연결이 느리면 시간이 좀 걸릴 수 있습니다.

[7] 하지만 3글자 약자 이름의 정보 기관 사람은 광자기 미디어를 여러 번 덮어 쓴다고 해도 데이터를 복구할 수 있다고 합니다.

베이스 시스템 설치 중에 패키지를 풀고 설정하면서 나오는 메시지는 **tty4**에서 표시합니다. 이 터미널은 왼쪽 Alt-F4을 누르면 이용할 수 있습니다. 설치 프로그램 화면으로 돌아가려면 왼쪽 Alt-F1을 누르십시오.

이 단계에서 압축을 풀고 설정하는 메시지는 /var/log/syslog 파일에 저장합니다. 시리얼 콘솔에서 설치하는 경우 이 파일을 보면 됩니다.

설치 도중에, Linux 커널을 설치합니다. 기본 우선순위에서는 설치 프로그램이 하드웨어에 가장 맞는 커널을 하나 선택합니다. 우선순위가 낮은 모드에서는, 사용 가능한 여러가지 커널 중에서 하나를 선택할 수 있습니다.

패키지 관리 시스템을 사용하여 패키지를 설치할 때 기본적으로 해당 패키지가 권장하는 패키지도 설치됩니다. 추천 패키지는 선택한 소프트웨어의 핵심 기능에 꼭 필요하지는 않지만, 패키지 관리자 의견으로는 일반적으로 그 소프트웨어와 같이 설치하면 기능이 확장되는 패키지입니다.

참고

기술적인 이유 때문에 베이스 시스템 설치 중에 설치되는 패키지는 그 패키지의 “권장” 패키지를 설치하지 않습니다. 위에서 말한 규칙은 베이스 시스템을 설치한 다음부터 적용됩니다.

6.3.6 추가 소프트웨어 설치하기

이제 사용할 수는 있을 만한 상태이지만 아주 제한적인 시스템이 됩니다. 이 시스템에 추가로 소프트웨어를 설치해서 사용자의 필요에 맞게 시스템을 맞출 수 있고, 설치 프로그램에서 그 작업을 합니다. 컴퓨터나 네트워크가 느리면 이 작업은 베이스 시스템 설치보다도 더 오래 걸릴 수 있습니다.

6.3.6.1 APT 설정하기

데비안 GNU/리눅스에서 패키지를 설치할 때 쓰는 프로그램의 하나는 apt 패키지에 있는 **apt**입니다. [8] **aptitude**나 **synaptic** 등의 패키지 관리 프로그램도 사용하는 사람이 많으며 이 프로그램은 **apt**에 의존합니다. 처음 사용자는 뒤에 언급한 프로그램을 사용하길 권장합니다. 패키지 검색 기능이나 상태 확인 등의 기능을 사용자 인터페이스에 내장하고 있기 때문입니다.

apt에서 패키지를 어디서 가져올 지 설정합니다. 설정한 내용은 /etc/apt/sources.list에 기록하고 설치가 끝난 다음 이 파일의 내용을 살펴보고 바꿀 수 있습니다.

기본값 우선순위에서 설치한다면, 설정 대부분을 설치 프로그램이 자동으로 처리합니다. 사용하는 설치 방법에 따라 자동으로 설정하기도 하고, 설치 앞 단계에서 지정한 사항에 맞춰서 자동으로 설정하기도 합니다. 설치 프로그램에서 자동으로 보안 미러 사이트를 추가하고, 안정 버전 배포판을 설치한다면 “stable-updates” 업데이트 서비스에 대한 미러 사이트도 추가합니다.

낮은 우선순위로 설치한다면(예를 들어 전문가 모드), 더 많은 결정을 직접 내릴 수 있게 됩니다. 보안 업데이트와 안정 업데이트 서비스를 사용할 지 여부를 지정할 수 있고, 아카이브의 “contrib” 및 “non-free” 섹션의 패키지를 추가할 지도 결정할 수 있습니다.

[8] 참고로 패키지를 실제로 설치하는 프로그램은 **dpkg**입니다. 하지만 이 프로그램은 저수준 도구에 가깝습니다. **apt**는 고수준 도구로 필요할 때 **dpkg**를 실행합니다. **apt**는 패키지를 설치 미디어, 네트워크 등 어디에서 가져올 지 판단합니다. 또 설치하려는 패키지가 제대로 동작하려면 필요한 다른 패키지까지 자동으로 설치합니다.

6.3.6.1.1 여러 개의 CD나 DVD 이미지에서 설치하기

큰 모음의 일부인 CD나 DVD 이미지에서 설치하는 경우 설치 프로그램에서 추가로 설치 미디어를 읽을 지 여부를 물어봅니다. 르허나 미디어가 더 있을 경우 더 읽어들이면 설치 프로그램에서 그 미디어에 들어 있는 패키지를 사용할 수 있습니다.

추가 미디어가 없어도 문제가 없습니다. 추가 미디어가 꼭 필요하지는 않습니다. 하지만 (다음 섹션에서 설명하는) 네트워크 미러도 사용하지 않는 경우 태스크에 해당하는 패키지를 설치할 수 없을 수도 있습니다.

참고

패키지는 CD나 DVD 이미지에 유명한 패키지의 순서로 포함되어 있습니다. 즉 대부분의 사람은 이미지 모음의 첫번째 이미지만 사용하고, 이미지 모음의 마지막 이미지에 들어 있는 패키지를 사용하는 사람은 거의 없습니다.

이 말은 즉, 전체 CD 모음을 구입하거나, 다운로드하거나, 굽는 일은 돈 낭비가 될 수 있습니다. 보통 CD 전체를 사용하는 일은 없기 때문입니다. 대부분의 경우 설치할 때는 3장에서 8장 정도의 CD로 설치한 다음 기타 패키지는 인터넷에서 미러 사이트를 이용해 설치하는 편이 좋습니다. DVD 모음의 경우에도 마찬가지입니다. 첫번째 DVD나 두번째 DVD만으로도 필요한 대부분의 패키지가 들어 있습니다.

설치 미디어 여러 장을 읽어들이는 경우, 드라이브에 들어 있는 미디어에 없는 패키지가 필요할 때마다 설치 프로그램에서 알려 줍니다. 같은 모음에 속한 디스크만 읽어들어야 하니 주의하십시오. 읽어들이는 순서는 중요하지 않지만, 오름차 순으로 읽어들이면 실패할 가능성이 줄어듭니다.

6.3.6.1.2 네트워크 미러 사용하기

설치할 때 보통 패키지 공급 용도로 네트워크 미러를 사용할 지 여부에 대한 질문을 받게 됩니다. 기본값을 사용해도 되지만 예외의 경우가 있습니다.

전체 CD/DVD 이미지를 사용해 설치하는 경우가 아니라면, 네트워크 미러를 사용해야 합니다. 네트워크 미러를 사용하지 않으면 설치를 마쳤을 때 아주 최소한의 시스템만 설치하게 될 수 있습니다. 하지만 인터넷 연결이 느리다면 설치 단계에서 **데스크톱** 태스크를 선택하지 않는 편이 좋습니다.

전체 CD 이미지 한 개로 설치하는 경우, 네트워크 미러 설정은 필요가 없긴 하지만 그래도 강력히 추천합니다. CD 한 장에는 아주 일부의 패키지만 들어갈 수 있기 때문입니다. 인터넷 연결이 느리다면 네트워크 미러를 여기에서 설정하지 말고, CD 이미지에 들어 있는 패키지를 이용해 설치를 마친 다음(새 시스템으로 다시 시작한 다음) 나머지 패키지를 설치하는 게 좋습니다.

DVD에서 설치하는 경우, 설치에 필요한 패키지는 첫번째 DVD에 모두 들어 있습니다. 네트워크 미러는 꼭 사용하지 않아도 됩니다.

네트워크 미러를 추가하는 좋은 점은, 설치할 때 업데이트를 사용할 수 있다는 부분입니다. CD/DVD 이미지를 만드는 동안 업데이트가 발생해서 포인트 릴리스에 포함되기도 합니다. 즉 설치할 시스템의 보안이나 안정성을 해치지 않고도 CD/DVD 모음을 더 오래동안 사용할 수 있습니다.

정리하면, 네트워크 미러를 사용하는 게 항상 좋습니다. 단 인터넷 연결이 좋지 않다면 네트워크 미러를 사용하지 않는 게 좋습니다. 패키지의 현재 버전이 설치 미디어에 들어 있다면, 설치 프로그램은 항상 설치 미디어에 들어 있는 패키지를 이용합니다. 그러므로 미러 사이트를 사용할 경우 다운로드할 데이터의 크기는

1. 다음 설치 단계에서 선택하는 태스크,

2. 그 태스크에 필요한 패키지,
3. 그 패키지 중에 사용 중인 설치 미디어에 들어 있는 패키지,
4. 설치 미디어에 들어 있는 패키지의 업데이트 버전이 있다면 미러에 있는 지 여부에 (일반 패키지 미러이든 보안 업데이트이든 안정 업데이트 미러이든 간에) 달려 있습니다.

그 의미는 이렇습니다. 네트워크 미러를 사용하지 않으려는 경우에도, 미러를 설정해 놓으면 보안 업데이트나 안정 업데이트가 있는 경우에 패키지를 인터넷에서 다운로드할 수 있습니다.

6.3.6.1.3 네트워크 미러 고르기

설치 중에 네트워크 미러를 사용하겠다고 선택했으면, 설치 프로그램의 앞 단계에서 선택한 국가 설정에 따라 미러의 목록이 표시됩니다. 기본값을 선택하면 대부분 잘 동작합니다.

제공되는 기본값은 deb.debian.org입니다. 이 미러는 그 자체가 미러 사이트가 아니고 최신으로 업데이트되고 속도가 빠른 다른 미러로 리다이렉트됩니다. 이 미러 사이트는 TLS (https 프로토콜) 및 IPv6를 지원합니다. 이 서비스는 데비안 시스템 관리 팀에서 (DSA, Debian System Administration) 관리합니다.

"수동으로 정보 입력하기"를 선택해 수동으로 미러를 지정할 수도 있습니다. 그 다음에 미러 호스트 이름과 (필요하다면) 포트 번호를 지정할 수 있습니다. URL 베이스 값을 써야 합니다. 그러므로 IPv6 주소를 쓸 때 주소 앞뒤로 각괄호를 "[2001:db8:1]"처럼 써야 합니다.

컴퓨터가 IPv6 전용 네트워크에 연결되어 있다면 (대다수의 사용자는 보통 이런 경우가 아님) 해당 국가의 기본 미러 사이트를 선택해도 동작하지 않을 수도 있습니다. 목록의 모든 미러 사이트는 IPv4로 연결할 수 있지만, 아주 일부만 IPv6로 연결할 수 있습니다. 미러 사이트의 연결 상태는 시간이 지나면서 달라질 수 있지만, 이 연결 상태에 대한 정보는 설치 프로그램 안에 들어 있지 않습니다. 해당 국가의 기본 미러 사이트에 IPv6 연결이 없는 경우 다른 미러를 시도해 보거나 "수동으로 정보 입력" 옵션을 선택하십시오. 그 다음에 "ftp.ipv6.debian.org"를 미러 이름으로 입력하십시오. 이 사이트는 IPv6에서 사용할 수 있습니다 (빠른 사이트는 아니더라도).

6.3.6.2 소프트웨어 선택 및 설치

설치 과정에서 소프트웨어를 추가로 선택해서 설치할 기회가 있습니다. 85593 개의 패키지에서 패키지를 일일이 선택하지 않고, 미리 정해진 소프트웨어의 묶음을 선택하고 설치하는 데 중점을 두고 있습니다. 그래야 설치를 빨리 마쳐서 컴퓨터를 다양한 용도로 사용할 수 있습니다.

태스크는 "데스크톱 환경", "웹서버", [9] 등 컴퓨터로 할 수 있는 여러가지 작업을 대략적으로 나타냅니다. D.2절에서 각 태스크마다 필요한 공간을 볼 수 있습니다.

설치하는 컴퓨터의 특징에 따라 미리 태스크를 선택한 경우도 있습니다. 그게 마음에 들지 않으면 태스크 선택을 해제할 수도 있습니다. 여기에서 태스크를 하나도 선택하지 않을 수도 있습니다.

[9] 이 목록을 표시할 때, 설치 프로그램은 **tasksel**을 실행할 뿐입니다. **tasksel**은 시스템 설치를 끝낸 다음에도 언제든지 패키지를 설치하고 지울 때 실행할 수 있습니다. **tasksel** 외에도 **aptitude**처럼 패키지 설치와 제거를 더 자세히 하는 프로그램을 사용할 수도 있습니다. 설치를 모두 마치고 특정 패키지를 찾으려면 **aptitude install 패키지** 명령을 실행하면 됩니다. 여기서 **패키지**는 찾으려는 패키지의 이름입니다.

작은 정보

설치 프로그램의 표준 사용자 인터페이스에서는, 스페이스바를 눌러서 태스크를 선택하고 해제할 수 있습니다.

참고

"데스크톱 환경" 태스크는 그래픽 데스크톱 환경을 설치합니다.

기본값으로 debian-installer에서는 데스크톱 환경을 설치합니다. 설치할 때 다른 데스크톱 환경을 선택할 수도 있습니다. 여러가지 데스크톱을 설치하는 것도 가능하지만, 일부 데스크톱 조합은 같이 설치할 수 없을 수도 있습니다.

이 기능은 원하는 원하는 데스크톱 환경에 필요한 패키지가 있을 경우에만 동작합니다. 전체 CD 이미지 1장으로 설치하면 필요한 패키지가 용량 한계 때문에 해당 CD 이미지에 없을 수도 있습니다. 그러한 경우 네트워크의 미러 사이트에서 필요한 패키지를 다운로드하게 됩니다. DVD 이미지나 기타 설치 방법을 이용하는 경우에는 이 방법으로 어떤 데스크톱 환경이든 설치하는데 문제가 없습니다.

다양한 서버 태스크는 다음과 같이 소프트웨어를 설치합니다. 웹 서버: apache2, SSH 서버: openssh.

"표준 시스템" 태스크는 "표준" 우선 순위의 모든 패키지를 설치합니다. 리눅스 및 유닉스 시스템에 보통 들어 있는 수많은 유틸리티가 여기에 포함됩니다. 이 태스크는 선택해 놓아야 합니다(무슨 일을 하고 있는지 알고 있고 정말 최소한의 시스템을 원하는 경우가 아니라면).

언어 선택을 할 때 "C"가 아닌 기본 로캘을 선택했다면, **tasksel**에서 그 로캘에 대한 지역화 태스크가 있는지 검사해서 관련된 지역화 패키지를 설치합니다. 해당 언어에서 사용하는 단어 목록 및 특별히 필요한 글꼴이 여기에 해당됩니다. 데스크톱 환경을 선택했다면 데스크톱 환경에 필요한 지역화 패키지도(이러한 패키지가 있다면) 설치합니다.

태스크를 선택했으면, 계속 단추를 누르십시오. 그러면 **apt**에서 해당 태스크에 들어 있는 패키지를 설치합니다. 프로그램 중에 사용자가 설정해야 하는 프로그램이 있으면 설치하는 중에 알려 줍니다.

데스크톱 태스크는 특히 매우 크므로 주의해야 합니다. 일반 CD-ROM과 CD-ROM에 없는 패키지가 들어 있는 미러를 같이 쓰는 경우, 네트워크에서 수많은 패키지를 받아 와야 할 수도 있습니다. 인터넷 연결이 느린 경우에는 받아오는 데 오래 걸릴 수도 있습니다. 패키지 설치를 일단 시작하면 취소하는 방법은 없습니다.

패키지가 CD-ROM 안에 들어 있는 경우에도, CD-ROM에 들어 있는 패키지보다 미러에 있는 패키지의 버전이 최신이면 미러에 있는 패키지를 받아 옵니다. 안정 버전 배포판을 설치하는 경우에는 주로 포인트 릴리스를(안정 버전 릴리스 업데이트) 한 다음에 이런 일이 발생합니다. 테스팅 배포판을 설치하는 경우 오래된 CD 이미지를 사용하면 이런 일이 발생합니다.

6.3.7 시스템을 부팅 가능하게 만들기

디스크 없는(diskless) 워크스테이션에 설치하는 경우, 로컬 디스크에서 부팅하는 건 당연히 불가능한 방법이므로 이 단계는 건너 뜁니다.

6.3.7.1 다른 운영 체제 검색

부트 로더가 설치되기 전에 설치 프로그램이 이미 설치되어 있는 다른 OS의 검색을 시도합니다. 지원하는 OS가 있으면 부트 로더 설치 단계 동안에 그것을 통지합니다. 또한 데비안 이외에 다른 OS를 부팅할 수 있도록 컴퓨터를 설정합니다.

한 컴퓨터에서 여러 개의 운영 체제를 부팅하는 일은 아직까지도 매우 복잡한 기술입니다. 다른 운영 체제를 자동으로 찾아내고 부트로더를 설정하는 일은 아키텍처마다 다르고, 서브 아키텍처마다 다르기도 합니다. 동작하지 않으면 부트로더의 문서에서 더 자세한 사항을 찾아보십시오.

6.3.7.2 시스템을 flash-kernel로 부팅 가능하게 만들기

ARM 플랫폼에서는 공통의 펌웨어 인터페이스가 없기 때문에, ARM 장치에서 시스템을 부팅시키는 단계는 장치에 따라 크게 다릅니다. 데비안은 **flash-kernel**라는 도구를 이 목적으로 사용합니다. flash-kernel에는 시스템을 여러가지 장치에서 부팅시키게 만드는 절차에 관한 데이터베이스가 들어 있습니다. flash-kernel은 현재 장치를 지원하는지 검색하고, 지원하는 경우 필요한 작업을 수행합니다.

내부의 NOR 또는 NAND 플래시 메모리에서 부팅하는 장치의 경우, flash-kernel에서 커널 및 최초 램디스크를 그 내부 메모리에 씁니다. 이 방식은 예전의 armel 장치에서 특히 많이 사용합니다. 대부분의 장치는 내부 플래시 메모리에 여러 개의 커널과 램디스크를 탑재하도록 지원하지 않기 때문에, flash-kernel을 이러한 장치에서 실행하면 이전의 플래시 메모리 내용을 덮어쓰게 됩니다!

시스템 펌웨어로 U-Boot를 사용해 커널과 최초 램디스크를 외부 저장 장치에서 (MMC/SD 카드, USB 저장 장치, IDE/SATA 하드 디스크 등) 부팅할 수 있는 ARM 시스템의 경우, flash-kernel에서는 사용자가 입력하지 않아도 자동 부팅할 수 있도록 적절한 부팅 스크립트를 만듭니다.

6.3.7.3 부트로더 없이 계속

이 옵션은 부트로더를 설치하지 말고 설치를 마칠 때 사용할 수 있습니다. 이렇게 하는 경우는 아마도 해당 아키텍처나 서브 아키텍처에 부트로더가 없거나, 부트로더가 필요없는 경우일(예를 들어 기존 부트로더를 사용) 것입니다.

부트로더를 수동으로 설정하려면, `/target/boot`에 설치한 커널의 이름을 확인해야 합니다. 또 이 디렉터리에 initrd가 있는 지 확인하고, 있으면 아마도 부트로더가 그 initrd를 사용하도록 해야 할 것입니다. 그 외에 / 파일 시스템으로 사용하려는 디스크 및 파티션을 알아야 하고, `/boot`가 별도 파티션이면 `/boot` 파일시스템의 디스크 및 파티션도 알아야 합니다.

6.3.8 설치 마치기

이제 설치 프로그램이 할 몇 가지 작업은 데비안 설치 과정에서 마지막 단계입니다. 대부분은 `debian-installer`의 뒷정리입니다.

6.3.8.1 시스템 시계 맞추기

설치 프로그램에서 컴퓨터의 시계를 UTC로 맞출 지 물어보기도 합니다. 다른 운영 체제를 설치했는 지 따위를 이용해 자동으로 UTC로 맞출 지 여부를 판단하기 때문에, 이 질문은 보통 물어보지 않습니다.

전문가 모드에서는 UTC로 맞출 지 여부를 항상 선택할 수 있습니다.

`debian-installer`가 여기에서 현재 시각을 시스템의 하드웨어 시계에 저장합니다. 앞에서 설정한 사항에 따라 UTC로 저장할 수도 있고 지역별 시각으로 저장할 수도 있습니다.

6.3.8.2 시스템 다시 시작

설치를 시작할 때 사용했던 부팅 미디어(CD, USB 메모리 등)를 검색 합니다. 그 다음에 새로 설치한 데비안 시스템으로 다시 시작합니다.

6.3.9 문제해결

여기에 목록이 나와 있는 컴포넌트는 일반적인 설치 과정과는 상관이 없습니다. 하지만 백그라운드에서 기다리면서 잘못된 부분이 있을 때 도움이 됩니다.

6.3.9.1 설치 로그 저장

설치가 성공적 이었으면, 설치할 때 만든 로그 파일은 새 데비안 시스템 `/var/log/installer/`에 자동으로 작성되고 있습니다.

메인 메뉴에서 디버깅 기록 저장을 선택하면 로그 파일을 USB 메모리, 네트워크, 하드디스크 등등의 미디어에 저장할 수 있습니다. 설치 도중 심각한 문제가 발생했을 경우 다른 시스템에서 로그를 분석하거나, 로그를 설치 보고서에 첨부할 때 유용합니다.

6.3.9.2 쉘 사용하기 및 로그 보기

설치하는 도중에 쉘을 실행하려면 여러가지 방법이 있습니다. 대부분의 시스템에서 시리얼 콘솔로 설치하는 게 아니라면, 가장 쉬운 방법은 왼쪽 Alt-F2를 눌러[10] (맥 키보드에서는 Option-F2) 두번째 가상 콘솔로 전환하는 것입니다. 왼쪽 Alt-F1을 누르면 설치 프로그램으로 다시 돌아올 수 있습니다.

콘솔을 전환할 수 없다면, 메인 메뉴의 쉘 실행 항목을 이용해도 쉘을 시작할 수 있습니다. 뒤로 가기 단추를 계속 눌러서 메인 메뉴로 돌아올 수 있습니다. 설치 프로그램으로 돌아오려면 **exit**를 입력해서 쉘을 닫으십시오.

램디스크에서 부팅했기때문에 이 쉘에서는 제한적인 유닉스 유틸리티만 사용할 수 있습니다. 어떤 프로그램이 있는지는 **ls /bin /sbin /usr/bin /usr/sbin** 명령 및 **help**를 입력해서 알 수 있습니다. 쉘은 **ash**이라고 하는 본 쉘 호환 쉘이고 자동 완성이나 명령어 기록같은 훌륭한 기능도 일부 들어 있습니다.

파일을 편집하거나 파일을 보려면, **nano** 텍스트 편집기를 사용하십시오. 설치 시스템의 로그 파일은 `/var/log` 디렉터리 안에 들어 있습니다.

참고

쉘에서는 실행할 수 있는 명령어로 무엇이든 할 수 있는 게 사실이지만, 쉘을 사용하는 옵션은 사실 뭔가 잘못된 경우에 대비해서 혹은 디버깅용으로 만들어 놓은 옵션입니다.

쉘에서 수동으로 무언가를 할 경우에 설치 과정이 방해를 받아서 오류가 발생하거나 설치를 끝마치지 못할 수도 있습니다. 특히 스왑 파티션을 활성화하는 기능은 설치 프로그램에서 알아서 하도록 놔두고 쉘에서 직접 하지 않도록 하십시오.

[10] **스페이스 바** 왼쪽에 있는 **Alt** 키와 펑션 키의 **F2**를 동시에 누르는 걸 말합니다.

6.3.10 네트워크 콘솔을 통해 설치

재미있는 컴포넌트 중의 하나로 network-console이 있습니다. 설치 작업의 많은 부분을 네트워크 SSH을 통해 수행하게 되어 있습니다. 네트워크를 사용해야 하기 때문에 최소한 네트워크 설정하기까지의 맨 처음 설치 작업은 콘솔에서 해야 합니다. (이 부분은 4.4절에 따라 자동화할 수 있습니다.)

이 컴포넌트는 주 설치 메뉴에는 기본으로 읽어들이지 않기 때문에, 이 컴포넌트를 읽어들이라고 지정해야 합니다. 광학 미디어에서 설치하는 경우에는 중간 우선 순위로 설치하거나 주 설치 메뉴가 나타나면 설치 미디어에서 설치 프로그램 컴포넌트를 읽어들이기을 선택하고 network-console: SSH을 사용해 원격에서 설치하기 추가 컴포넌트를 선택합니다. 성공적으로 읽어들이면 SSH을 사용해 원격에서 설치하기 메뉴 항목이 새로 생깁니다.

에서 새 항목을 선택한 후 설치 시스템에 연결하기 위한 새 암호(및 확인)를 입력하십시오. 여기까지 하면 지금 원격 로그인하라는 화면이 나옵니다. 사용자 이름은 installer, 방금 입력한 암호를 사용하십시오. 이 화면에 있는 중요한 정보로 이 시스템의 핑거프린트가 있습니다. 이 핑거프린트를 원격에서 설치할 사람에게 안전하게 전달해야합니다. 새로 생긴 이 항목을 선택한 다음에, 설치 시스템에 연결하는데 사용할 새 암호를 입력하게 됩니다. 여기까지 하면 installer 사용자로 방금 입력한 암호를 이용해 원격에서 로그인할 수 있는 방법을 알려주는 화면을 표시합니다. 이 화면의 또 다른 중요한 정보는 시스템의 핑거프린트입니다. 이 핑거프린트를 “시스템을 원격에서 설치할 사람에게” 안전하게 전달해야 합니다.

로컬에서 설정을 계속하려고 마음을 바꿨다면, 언제든지 **Enter**를 눌러서 메인 메뉴로 돌아갈 수 있습니다. 메인 메뉴에서 다른 컴포넌트를 선택하면 됩니다.

이제 네트워크의 다른 한 편으로 가서 할 일입니다. 먼저 필요한 일은, 터미널을 UTF-8 인코딩을 쓰도록 설정하는 일입니다. UTF-8 인코딩이 설치 시스템에서 사용하는 인코딩입니다. UTF-8으로 설정하지 않아도 원격 설치가 가능하지만 창의 테두리라던지, 읽을 수 없는 ASCII가 아닌 문자처럼 표시가 깨질 수도 있습니다. 설치 시스템에 연결하려면 간단히 다음과 같은 명령을 사용하면 됩니다:

```
$ ssh -l installer install_host
```

여기서 `install_host`는 설치할 컴퓨터의 이름이나 IP 주소입니다. 실제로 로그인하기 전에 원격 시스템의 핑거프린트가 표시될 것이고, 이 핑거프린트가 올바른 지 확인해야 합니다.

참고

설치 프로그램에 들어 있는 **ssh** 서버는 연결 유지(keep-alive) 패킷을 보내지 않는 표준 설정을 사용합니다. 시스템에 대한 연결은 계속해서 열어 놓은 상태여야 합니다. 하지만 (로컬 네트워크 설정에 따라) 일정 시간동안 아무 입력이 없으면 연결이 끊어질 수 있습니다. 이런 일이 벌어질 수 있는 흔한 상황이 SSH 클라이언트와 설치하는 서버 사이 어딘가에 NAT(Network Address Translation, 네트워크 주소 변환)가 있는 상황입니다. 연결이 어느 부분에서 끊어졌느냐에 따라 다시 연결했을 때 설치를 계속 할 수도 있고 할 수 없을 수도 있습니다.

ssh 연결을 시작할 때 **-o ServerAliveInterval=값** 옵션을 사용하면 연결이 끊어지는 일을 방지할 수 있습니다. 아니면 이 옵션을 **ssh** 설정 파일에 추가해도 효과가 같습니다. 하지만 이 옵션을 사용했을 때 이 옵션때문에 연결이 끊어지는 경우도 있습니다. (예를 들어 연결 유지(keep-alive) 패킷을 일시적인 네트워크 정지 시점에 보내는 경우 그렇습니다. 이러한 경우 **ssh** 연결은 다른 방법으로 복구합니다.) 그러니 이 옵션은 필요한 경우에만 사용해야 합니다.

참고

여러 컴퓨터를 모두 설치하는 경우에 IP 주소 혹은 호스트 이름이 같은 경우, 그런 호스트는 **ssh**에서 연결을 거부합니다. 그 이유는 핑거프린트가 다르기 때문이고, 핑거프린트가 다르다는 건 스푸핑 공격의 징조입니다. 스푸핑이 아니라고 확신하는 경우, `~/.ssh/known_hosts`에서 해당 줄을 지우고[a] 다시 연결하면 됩니다.

[a] 다음 명령어로 호스트의 해당 줄을 지울 수 있습니다: **ssh-keygen -R <호스트이름|IP 주소>**.

로그인한 다음에 최초 화면이 나오면 거기에서 메뉴 시작과 쉘 시작 중의 하나를 선택할 수 있습니다. 전자의 경우는 설치 프로그램의 메인 메뉴로 가게 되고, 거기에서 로컬에서와 마찬가지로 설치 작업을 계속할 수 있습니다. 후자는 쉘을 실행해서 원격 시스템을 살펴보고 문제점을 수정할 수 있습니다. 설치 메뉴는 한 개의 SSH 세션만 열어야 합니다. 하지만 쉘의 경우에는 여러 개를 열어도 됩니다.

주의

SSH을 통해 원격으로 설치를 시작하면, 다시 로컬 콘솔로 돌아가서 설치하면 안 됩니다. 그렇게 하면 새로 설치할 시스템의 설정이 망가질 수 있습니다. 설정이 망가지면 설치가 실패하거나 새로 설치한 시스템에 여러가지 문제가 발생할 수 있습니다.

6.4 없는 펌웨어 읽어들이기

2.2절에서 설명한 것처럼, 일부 장치는 펌웨어를 읽어들여야 합니다. 대부분 그런 장치는 펌웨어가 없으면 동작하지 않습니다. 가끔 기본적인 기능은 동작하고 추가적인 기능에서만 펌웨어가 필요하기도 합니다.

없는 펌웨어가 장치 드라이버에 필요한 경우, debian-installer에서 대화 상자를 표시해 없는 펌웨어를 읽어들이도록 안내합니다. 이 옵션을 선택하면 debian-installer는 펌웨어 파일이나 펌웨어가 들어 있는 패키지가 있는 지 검사합니다. 있으면 펌웨어를 적당한 위치에 (/lib/firmware) 복사하고 드라이버 모듈을 다시 읽어들입니다.

참고

어떤 장치를 검사하고 어떤 파일 시스템을 지원하느냐는 아키텍쳐, 설치 방법, 설치 단계에 따라 달라집니다. 설치 앞 단계에서는 FAT로 포맷한 USB 메모리에서 펌웨어 읽어들이기는 대부분 성공합니다.

펌웨어 없이도 해당 장치가 동작한다거나 설치할 때 그 장치가 필요 없는 경우, 펌웨어 읽어들이기를 건너뛸 수도 있습니다.

debian-installer에서는 설치 중에 올라간 커널 모듈에서 필요한 펌웨어만 요청합니다. 모든 드라이버가 debian-installer에 포함되지는 않았기 때문에(특히 radeon이 없음), 일부 장치는 설치가 끝날 때 쯤 되어도 설치가 시작될 때나 다를 것 없이 사용되지 않을 수도 있습니다. 결과적으로 하드웨어의 일부는 완전히 사용할 수 없을 수도 있습니다. 이런 상황이 의심되거나 궁금하다면 새로 부팅하는 시스템의 **dmesg** 명령 결과에서 “firmware”라는 말을 찾아보십시오.

6.4.1 미디어 준비하기

공식 설치 이미지에는 사용 제한이 있는 펌웨어는 들어 있지 않습니다. 이러한 펌웨어를 읽어들이는 방법 중 가장 많이 사용하는 방법은 USB 메모리같은 이동식 장치에서 펌웨어를 읽어들이는 경우입니다. 아니면 사용 제한이 있는 펌웨어가 들어있는 비공식 설치 이미지 빌드가 https://cdimage.debian.org/cdimage/unofficial/non-free/cd-including-firmware/ 위치에 있습니다. USB 메모리(아니면 하드 드라이브 파티션 등 다른 미디어)를 준비하려면, 펌웨어 파일이나 패키지는 해당 미디어의 최상위 디렉터리나 /firmware 디렉터리에 들어 있어야 합니다. 추천하는 파일 시스템은 FAT입니다. (FAT는 설치 앞 단계에서도 지원하는 파일 시스템이므로.)

많이 사용하는 펌웨어 패키지를 tar나 zip으로 굳힌 것이 다음의 사이트에서 사용할 수 있습니다:

- https://cdimage.debian.org/cdimage/unofficial/non-free/firmware/

해당 버전의 tarball 또는 zip 파일을 다운로드 미디어 파일 시스템에 배포하면됩니다.

필요한 펌웨어가 TAR 파일에 없으면 해당 펌웨어 패키지를 아카이브에서(아마도 non-free 섹션에서) 다운로드할 수도 있습니다. 아래에서 흔히 사용하는 펌웨어 패키지 목록을 요약해 놓았습니다. 아래 목록은 전체 목록이 아니고 펌웨어 패키지가 아닌 패키지도 들어 있습니다:

- https://packages.debian.org/search?keywords=firmware

펌웨어 파일을 직접 미디어에 복사할 수도 있습니다. 펌웨어 파일을 이미 설치해 놓은 시스템에서 가져올 수도 있고 하드웨어 공급사가 제공해 줄 수도 있습니다.

6.4.2 펌웨어 및 설치한 시스템

설치할 때 읽어들인 펌웨어는 설치한 시스템에도 복사됩니다. 그래야 펌웨어가 필요한 장치가 설치한 시스템으로 다시 시작한 후에도 올바르게 동작합니다. 하지만 설치한 시스템의 커널 버전이 다른 경우 펌웨어를

읽어들이지 못할 가능성이 낮지만 있습니다.

펌웨어를 펌웨어 패키지에서 읽어들인 경우, debian-installer는 이 패키지를 설치한 시스템에서도 설치하고 APT의 sources.list에 non-free 섹션을 추가합니다. 이렇게 하면 펌웨어의 새 버전이 나왔을 때 자동으로 업데이트하는 장점이 있습니다.

설치할 때 펌웨어 읽어들이기를 건너 뛰면, 수동으로 펌웨어를 설치하기 전에는 해당 장치가 설치한 시스템에서 동작하지 않습니다.

참고

펌웨어 파일에서 펌웨어를 설치한 경우, 그 펌웨어는 설치한 시스템에 복사되고 해당 펌웨어 패키지를(패키지가 있다면) 설치하지 않는 한 자동으로 업데이트되지 않습니다.

6.4.3 시스템 설치 마치기

설치를 어떻게 수행했냐에 따라, 설치 중에 일부 필요한 펌웨어를 찾지 못했을 수도 있고, 관련 펌웨어가 없을 수도 있고, 그 시점에 펌웨어 설치를 하지 않겠다고 선택했을 수도 있습니다. 어떤 경우에는 성공적으로 설치했다고 해도 설치한 시스템으로 재시작했을 때 검은색 화면이나 알아 보기 힘든 화면만 나타나기도 합니다. 그런 일이 생기면, 다음 피해가는 방법을 시도해 볼 수 있습니다:

- 커널 커맨드라인에 nomodeset 옵션을 전달합니다. 그러면 "대비책 그래픽" 모드로 부팅될 수도 있습니다.
- Ctrl-Alt-F2 키 조합을 사용해 VT2로 전환하빈다. 그러면 동작하는 로그인 프롬프트가 나타날 수도 있습니다.

설치한 시스템으로 일단 로그인하면, 찾지 못했던 펌웨어 검색을 자동으로 할 수 있습니다. 다음 과정을 따라 펌웨어를 사용하는데 필요한 단계를 할 수 있습니다.

1. isenkram-cli 패키지를 설치합니다.
2. "root" 사용자로 **isenkram-autoinstall-firmware** 명령을 실행합니다.

확실하게 모든 커널 모듈을 올바르게 초기화하려면, 보통 다시 시작하는 게 가장 간단하게 할 수 있는 방법입니다. 특히 nomodeset 옵션을 중간 단계로 사용해 부팅한 시스템에서는 중요합니다.

참고

펌웨어 패키지 설치는 패키지 저장소의 non-free 섹션을 사용해야 할 경우가 대부분입니다. 데비안 GNU/리눅스 11.0에서는, **isenkram-autoinstall-firmware** 명령을 실행하면 미러 사이트를 가리키는 전용 파일을 (/etc/apt/sources.list.d/isenkram-autoinstall-firmware.list) 만들어서 해당 작업을 자동으로 해 줍니다.

6.5 원하는대로 바꾸기

셸을 사용해 (6.3.9.2절 참고), 설치 과정을 조심스럽게 원하는대로 (특히 예외적인 사용처에 맞출 때) 바꿀 수 있습니다.

6.5.1 다른 init 시스템 설치하기

데비안은 기본 init 시스템으로 systemd를 사용합니다. 하지만, 다른 init 시스템도 (sysvinit 및 OpenRC 등) 지원합니다. 다른 init 시스템을 선택할 수 있는 가장 좋은 때는 설치 과정 중입니다. 어떻게 하는지에 대한 안내는, 데비안 위키의 Init 페이지를 참고하십시오.

제 7 장

새로운 데비안 시스템으로 부팅하기

7.1 진실의 시간

시스템이 혼자 힘으로 하는 최초의 부팅을 전기 엔지니어는 "스모크 테스트(smoke test)"라고 부릅니다.

시스템이 제대로 시작하지 않았다하더라도 당황하지 마십시오. 설치가 성공적으로 완료됐다면 시스템이 데비안을 시작하는 것을 방해하는 비교적 작은 문제가 있을 가능성이 높습니다. 대부분의 경우 그런 문제는 보통 다시 설치하지 않아도 해결할 수 있습니다. 부팅시 문제를 해결하는 한 가지 방법은 설치 프로그램에 내장된 응급 복구 모드(8.6절 참조)를 사용하는 것입니다.

만약 데비안 및 Linux에 생소한 경우에, 경험있는 사용자의 도움이 필요 할지도 모릅니다. 32-bit hard-float ARMv7처럼 많이 사용하지 않는 아키텍처의 경우는 debian-arm 메일링 리스트 를 이용하는 게 가장 좋은 방법입니다. 5.4.5절를 따라 설치 보고서를 제출할 수도 있습니다. 보고서는 문제를 명확하게 설명 표시된 모든 메시지를 넣어 다른 사람이 문제의 원인을 파악되도록 하십시오.

7.2 암호화 볼륨 마운트하기

설치할 때 암호화 볼륨을 만들고 마운트 위치를 지정했다면, 부팅할 때 각각의 볼륨에 대해 암호를 입력하게 됩니다.

dm-crypt로 암호화한 파티션의 경우 부팅할 때 다음과 같이 물어봅니다:

```
Starting early crypto disks... part_crypt(starting)
Enter LUKS passphrase:
```

첫번째 줄에서, part는 실제 파티션의 이름입니다. (예를 들어 sda2나 md0.) 여기에서 과연 어떤 볼륨의 암호를 실제로 입력해야 하는 지 의문이 들 것입니다. /home 아니면 /var 일까요? 물론, 암호화 볼륨이 1개뿐이라면, 이 볼륨을 설정할 때 사용한 암호를 입력하면 됩니다. 설치할 때 암호화 볼륨을 여러 개 설정했다면, 6.3.4.6절의 마지막 단계에서 적어 놓은 메모를 잘 가지고 있어야 합니다. part_crypt에 해당되는 사항과 거기에 해당하는 마운트 위치를 적어 놓지 않았다면 새로 설치한 시스템의 /etc/crypttab과 /etc/fstab 에서 찾아 볼 수도 있습니다.

이 프롬프트는 암호화한 루트 파일시스템을 마운트할 때는 약간 다릅니다. 시스템을 부팅할 때 사용하는 initrd를 만들 때 어떤 initramfs 만들기 프로그램을 사용했느냐에 따라 다릅니다. 아래의 예제는 initramfs-tools로 initrd를 만들 경우에 대한 예제입니다:

```
Begin: Mounting root file system... ...
```

```
Begin: Running /scripts/local-top ...
Enter LUKS passphrase:
```

암호를 입력할 때는 아무런 글자도(별표 조차도) 나타나지 않습니다. 암호를 잘못 입력하면 두 번 더 시도할 수 있습니다. 세 번째 시도에서 틀리면 부팅 과정에서 해당 볼륨을 건너뛰고 다음 파일 시스템으로 넘어갑니다. 자세한 정보는 7.2.1절 부분을 보십시오.

암호를 모두 입력하면 부팅은 평소처럼 계속 진행합니다.

7.2.1 문제 해결

암호가 틀려서 암호화 볼륨을 마운트하지 못할 경우, 부팅한 다음에 수동으로 마운트해야 합니다. 여러가지 경우가 있습니다.

- 첫번째 경우는 루트 파티션입니다. 올바르게 마운트하지 않으면, 부팅 과정이 멈추게 되고 컴퓨터를 다시 시작해서 암호를 다시 입력해야 합니다.
- 가장 쉬운 경우는 /home이나 /srv처럼 데이터가 들어 있는 암호화 볼륨입니다. 부팅하고 수동으로 볼륨을 마운트하면 됩니다.

  하지만 dm-crypt의 경우에는 약간 까다롭습니다. 먼저 해당 볼륨을 다음 명령어로 device mapper에 등록해야 합니다:

  ```
  # /etc/init.d/cryptdisks start
  ```

  이렇게 하면 /etc/crypttab에 들어 있는 모든 볼륨을 검색하고 암호를 올바르게 입력할 때마다 /dev 디렉터리 아래에 적당한 장치를 만듭니다. (이미 등록한 볼륨은 건너 뛰므로, 걱정하지 말고 이 명령어를 여러번 실행해도 됩니다.) 올바르게 등록을 마치면 해당 볼륨을 평소와 다름없이 마운트할 수 있습니다:

  ```
  # mount /마운트_위치
  ```

- 꼭 필요하지는 않은 시스템 파일이 들어 있는 볼륨중에 하나라도(/usr 혹은 /var) 마운트할 수 없는 경우, 그래도 시스템이 부팅하고 수동으로 볼륨을 마운트할 수 있습니다. 하지만 현재 런레벨의 각종 서비스를(다시) 시작해야 할 수도 있습니다. 서비스가 제대로 시작하지 않았을 가능성이 높기 때문입니다. 가장 쉬운 방법은 컴퓨터를 다시 시작하는 것입니다.

7.3 로그인

패키지 설치가 끝나면 로그인 프롬프트를 표시합니다. 설치할 때 입력한 개인 로그인 및 암호를 이용해 로그인합니다. 그러면 이제 시스템을 사용할 준비를 다 마쳤습니다.

처음 설치한 사용자라면 문서를 살펴보고 싶을 것이고, 이 문서는 시스템을 시작할 때부터 시스템 안에 설치되어 있습니다. 현재 여러 개의 문서 시스템이 있고, 여러가지 종류의 문서를 통합하는 작업을 진행하고 있습니다. 다음과 같은 방법으로 문서 보기를 시작할 수 있습니다.

설치한 프로그램에 들어 있는 문서는 /usr/share/doc/ 아래에, 그 프로그램의 이름으로(정확히 말해 그 프로그램이 들어 있는 데비안 패키지의 이름으로) 된 서브 디렉터리에 들어 있습니다. 하지만 이보다 자세한 문서는 별도의 문서 패키지에 들어 있고, 이 패키지는 보통 기본으로 설치하지 않습니다. 예를 들어 **apt** 패키지 관리 도구에 관한 문서는 apt-doc 혹은 apt-howto 패키지에 들어 있습니다.

또 /usr/share/doc/ 아래에 특수 폴더가 몇 개 더 있습니다. 리눅스 HOWTO는 /usr/share/doc/HOWTO/en-txt/ 안에 .gz (압축한) 형식으로 들어 있습니다. dhelp를 설치하면 /usr/share/doc/HTML/index.html 파일에 브라우저로 볼 수 있는 문서 목록이 있습니다.

다음 명령으로 텍스트 기반 브라우저를 사용하면 간단히 이 문서를 볼 수 있습니다 :

```
$ cd /usr/share/doc/
$ w3m .
```

w3m 명령 다음에 나오는 점은 현재 디렉터리의 내용을 표시한다는 뜻입니다.

그래픽 데스크톱 환경을 설치했다면, 그 환경의 웹 브라우저를 이용할 수 있습니다. 프로그램 메뉴에서 웹 브라우저를 실행해서 주소창에 **/usr/share/doc/**을 입력하고 Enter를 누르십시오.

info 명령어 또는 **man 명령어** 명령을 입력하면, 명령어 프롬프트에서 사용할 수 있는 대부분의 명령에 대한 문서를 볼 수 있습니다. **help**를 입력하면 쉘 명령어에 대한 도움말을 표시합니다. 명령어 뒤에 **--help** 옵션을 붙이면 짤막한 명령어 사용법을 표시합니다. 명령어의 결과가 화면 위로 지나가 버린다면 **| more**를 명령 뒤에 붙이면 화면 위로 스크롤되 지나가기 전에 출력을 일시 정지할 수 있습니다. 어떤 글자로 시작하는 명령어의 목록을 보려면 그 글자를 입력하고 탭을 두번 누릅니다.

제 8 장

다음 단계 및 그 다음에 할 일

8.1 시스템 끄기

실행중인 데비안 GNU/리눅스 시스템을 종료할 때 컴퓨터의 앞이나 뒤에 있는 리셋 스위치를 눌러서 다시 시작하거나, 전원을 꺼 버려서는 안됩니다. 데비안 GNU/리눅스는 적절한 절차로 종료해야 하고, 그렇지 않으면 파일이 지워지거나 디스크에 손상이 올 수 있습니다. 데스크톱 환경을 실행하는 경우 정상적인 시스템 종료(또는 다시 시작)를 허용하는 응용 프로그램 메뉴에서 사용할 수 "로그아웃"의 옵션이 있습니다.

다른 방법으로 Ctrl-Alt-Del 키 조합을 누를 수도 있습니다. 이 키 조합이 동작하지 않으면, 마지막 방법은 root로 로그인해서 필요한 명령어를 입력하는 방법입니다. **reboot**로 시스템을 리부팅합니다. 전원을 끄지 않고 **halt**로 시스템을 멈춥니다[1]. 컴퓨터의 전원을 끄려면 **poweroff** 또는 **shutdown -h now** 명령을 사용합니다. systemd init 시스템은 같은 기능을 하는 다른 명령어가 있습니다. 예를 들어 **systemctl reboot** 또는 **systemctl poweroff** 명령어를 쓸 수 있습니다.

8.2 데비안에 익숙해지기

데비안은 다른 배포판들과 약간 다릅니다. 다른 배포판에서 리눅스에 잘 아는 분도 시스템을 최상의 상태로 유지하기 위해서는 데비안에 대해 알아 두어야 할 수 있습니다. 이 장에서는 데비안에 익숙해지기 위해 도움이 되는 자료를 소개합니다. 데비안의 사용법을 일일이 설명하지는 않고, 성급한 사람을 위해 시스템 개요만 설명합니다.

8.2.1 데비안 패키지 시스템

알아야 할 가장 중요한 개념은 데비안 패키지 시스템이 있습니다. 기본적으로 시스템의 대부분은 패키지 시스템 관리하에 있습니다. 이 패키지 시스템에서 관리하는 디렉터리는:

- `/usr` (`/usr/local`를 제외)
- `/var` (`/var/local`을 만들고 다음의 디렉터리를 자유롭게 사용하는 것은 가능합니다)
- `/bin`

[1] SysV init 시스템에서는 **halt**는 **poweroff**와 같은 효과를 냈지만, (jessie 이후 기본인) systemd를 init 시스템으로 사용할 경우 그 효과가 다릅니다.

- /sbin
- /lib

예를 들어, /usr/bin/perl을 작동한 파일로 대체해도 동작에는 문제가 없고, 나중에 perl 패키지를 업데이트하면 여러분이 설정한 파일은 패키지로 대체하게 됩니다. 이것을 방지하려면 **aptitude**에서 패키지를 "hold" 하는 작업을 합니다.

APT는 가장 훌륭한 패키지 설치 도구의 하나입니다. 명령행 방식의 **apt**을 사용할 수도 있고, aptitude 또는 synaptic(**apt**의 그래픽 프론트엔드)을 사용할 수도 있습니다. APT를 이용해 main, contrib, non-free 모두에서 설치할 수 있습니다. 즉 데비안 GNU/리눅스의 패키지는 물론 동시에 제한적인 소프트웨어 패키지도 (엄격히 말해 데비안에 속해 있지 않다고 말하지만) 설치할 수 있습니다.

8.2.2 데비안용 추가 소프트웨어

기본 데비안 설치에 빠진 공식 및 비공식 소프트웨어 저장소가 있습니다. 여기에는 많은 사람이 중요하다고 생각하는 소프트웨어가 들어 있습니다. 이 추가 저장소에 대한 정보는 데비안 위키의 The Software Available for 데비안's Stable Release 페이지에 있습니다.

8.2.3 프로그램 버전 관리

같은 이름의 프로그램이 여러가지 버전이 있는 경우 update-alternatives에서 관리합니다. 여러 버전의 프로그램을 관리하고 있다면, update-alternative 맨페이지를 읽어 보십시오.

8.2.4 CRON 작업 관리

시스템 관리자 권한으로 하는 작업은 설정 파일이므로, 모두 /etc 안에 들어 있어야 합니다. 루트 권한으로 매일, 매주, 매달 실행할 CRON 작업이 있으면, 그 스크립트를 /etc/cron.{daily,weekly,monthly} 아래에 넣으십시오. 이 스크립트는 /etc/crontab에서 실행하고, 알파벳 순서로 하나씩 실행합니다.

한편 (1) 특정 사용자로 실행할 CRON 작업이 있거나 (2) 특정 시간이나 특정 주기로 실행할 작업이 있으면 /etc/crontab을 사용하거나, 아니면 더 좋은 방법으로 /etc/cron.d/아무개를 사용할 수 있습니다. 이 파일에는 CRON 작업을 실행할 사용자를 지정하는 필드가 따로 있습니다.

어떤 방법을 사용하든 파일을 편집하기만 하면 CRON에서 자동으로 인식하고 처리합니다. 다른 명령어를 실행할 필요가 없습니다. 더 자세한 정보는 cron(8), crontab(5), /usr/share/doc/cron/README.Debian 파일을 참고하십시오.

8.3 그 외의 읽을 거리 및 정보

Debian 웹사이트에는 데비안에 관한 많은 문서가 있습니다. 특히, Debian GNU/Linux FAQ와 Debian 참조를 참조하십시오. Debian 문서 프로젝트에는 데비안 문서에 대한 더 많은 인덱스가 포함되어 있습니다. 데비안의 커뮤니티에 사용자가 서로 지원하고 있습니다. 데비안의 메일링 리스트에 가입하려면 메일링 리스트 가입 페이지를 참조하십시오. 마지막으로 Debian 메일링리스트 아카이브에는 데비안에 관한 수많은 정보가 포함되어 있습니다.

특정 프로그램에 대한 정보를 보려면, **man 프로그램** 명령을 실행해 보시고, 아니면 **info 프로그램** 명령을 실행해 보십시오.

/usr/share/doc에 유용한 문서가 많이 있습니다. 특히, /usr/share/doc/HOWTO과 /usr/share/doc/FAQ에 흥미로운 정보가 많이 있습니다. 버그를 보고하려면 /usr/share/doc/debian/bug*을 참조하십시오. 특정 프로그램에 대한 데비안 특정 문제에 대해 읽으려면 /usr/share/doc/(패키지)/README.Debian을 참조하십시오.

GNU/리눅스에 관한 정보는 보통 Linux Documentation Project에 보면 있습니다. 여기에 GNU/리눅스 시스템에 관한 하우투 및 다른 훌륭한 정보가 들어 있는 링크가 있습니다.

리눅스는 유닉스를 구현한 것입니다. Linux Documentation Project (LDP)에 리눅스와 관련된 여러 가지 HOWTO 문서와 온라인 서적이 있습니다.

유닉스를 처음 접한다면, 책을 사서 읽어 보는 게 좋을 수도 있습니다. list of Unix FAQs에는 기억에 남을 만한 훌륭한 참고자료로 여러가지 유즈넷 문서 목록이 들어 있습니다.

8.4 시스템에 전자메일 준비하기

오늘날, 전자 메일은 많은 사람의 삶의 중요한 일부가 되어 있습니다. 전자 메일을 사용할 수 있도록 설정하기 전에 전자메일을 설정을 선택하고 정확하게 설정되었는지가 중요합니다. 데비안 유틸리티에서 기본적인 사항을 설명합니다.

전자메일 시스템은 크게 세 가지로 구성됩니다. 우선 사용자가 실제로 전자메일을 작성하고 읽는데 사용하는 Mail User Agent(MUA)가 있습니다. 그리고 한 컴퓨터에서 다른 컴퓨터로 전자메일을 전달하는 Mail Transfer Agent(MTA)가 있습니다. 마지막으로 받은 전자메일을 사용자의 편지함으로 전달하는 일을 하는 Mail Delivery Agent(MDA)가 있습니다.

이 세가지 기능은 각각 다른 프로그램에서 담당할 수도 있고, 한 개나 두 개 프로그램에서 담당할 수도 있습니다. 또한 전자메일 종류별로 여러가지 프로그램이 담당할 수도 있습니다.

리눅스 및 유닉스 시스템에서는 전통적으로 **mutt**가 아주 널리 쓰이는 MUA입니다. 전통적인 리눅스 프로그램 대부분과 마찬가지로 텍스트 기반 프로그램입니다. **mutt**는 보통 **exim**이나 **sendmail**을 MTA로 쓰고 **procmail**을 MDA로 씁니다.

그래픽 데스크톱 시스템의 증가와 인기로, GNOME의 **evolution**, KDE의 **kmail** 또는 Mozilla의 **thunderbird** 같은 그래픽 전자 메일 프로그램 사용이 더 일반적으로 되어 있습니다. 이 프로그램에는 MUA, MTA 및 MDA의 기능이 결합되어 있지만, 기존의 Linux 도구와 — 함께 사용할 수 — 있습니다.

8.4.1 기본 전자메일 설정

그래픽 메일 프로그램을 사용한다고 해도, 데비안 GNU/리눅스 시스템 전통적인 MTA/MDA를 설치하고 정확하게 설정하면 좋습니다. 시스템에서 동작하는 여러가지 유틸리티 [2] 시스템 관리자에게 문제(또는 잠재적인 문제)와 바뀐 사항을 알릴 때 전자 우편으로 중요한 알림을 보낼 수 있기 때문입니다.

이렇게 하려면 exim4 및 mutt 패키지를 설치할 수 있고(**apt install exim4 mutt** 명령 사용), 용량이 작지만 매우 유연한 MTA/MDA 조합입니다. 하지만 기본값으로 시스템의 로컬 메일만 처리하도록 설정되어 있고 시스템 관리자에게 (루트 계정) 보내는 메일은 설치할 때 만드는 사용자 계정으로 배달합니다[3].

시스템 전자메일을 배달할 때 그 메일은 /var/mail/계정_이름 파일 뒤에 추가됩니다. 해당 전자메일은 **mutt**로 읽을 수 있습니다.

[2] 예를 들어: **cron**, **quota**, **logcheck**, **aide**, ...

[3] 루트 메일을 전달하는 계정은 /etc/aliases에 설정되어 있습니다. 물론 일반 사용자 계정을 만들지 않았다면 메일은 루트 계정 자체에 배달합니다.

8.4.2 시스템 외부에 전자메일 보내기

앞에서 말한 것처럼, 설치한 데비안 시스템은 시스템 내부의 전자 메일을 처리하도록 설정되어 있고, 타인에게 메일을 보내거나 외부에서 메일을 받도록 설정되지 않습니다.

exim4에서 외부 전자메일을 처리하도록 하려면, 다음의 기초 설정 옵션을 참고하십시오. 테스트 메일을 올바르게 보내고 받는 지 확인하십시오.

그래픽 메일 프로그램을 사용하고 여러분의 인터넷 서비스 회사 혹은 여러분의 회사의 메일 서버를 사용한다면, 외부 전자메일을 처리하려고 exim4를 설정할 필요가 전혀 없습니다. 사용하려는 그래픽 메일 프로그램을 설정해서 전자메일을 보내고 받는 데 이용할 서버를 올바르게 설정하십시오. (이 설정 방법은 이 안내서의 범위를 벗어납니다.)

그러나, 그 경우에는 올바르게 전자 메일을 보낼 수 있도록 각 유틸리티를 설정해야 할지도 모릅니다. 그러한 유틸리티 하나는 데비안 패키지에 대한 버그 보고서를 보내는 기능을 하는 프로그램인 **reportbug**가 있습니다. 기본적으로 버그 리포트를 제출하려면 exim4를 사용할 수 있습니다.

reportbug가 외부 메일 서버를 사용하도록 설정하려면, **reportbug --configure** 명령을 실행해서 MTA가 있는 지 여부를 묻는 질문에 "no"라고 답하십시오. 그러면 그 다음에 버그 보고를 보낼 때 사용할 SMTP 서버를 입력할 수 있습니다.

8.4.3 Exim4 MTA 설정하기

시스템에서 외부 전자메일까지 처리하게 만드려면, exim4 패키지를 다시 설정해야 합니다[4]:

```
# dpkg-reconfigure exim4-config
```

명령어를 실행하면(루트 권한으로), 설정 파일을 작은 파일로 나눌 지 여부를 물어봅니다. 잘 모르겠으면 기본 옵션을 선택하십시오.

그 다음에 자주 사용하는 여러 가지 전자메일 시나리오를 표시합니다. 필요한 사항에 가장 가까운 시나리오를 선택하십시오.

인터넷 사이트 시스템이 네트워크에 연결되어 있고 메일은 SMTP를 사용해 직접 보내고 받습니다. 다음 화면에서 시스템의 메일 이름이나 사용할 도메인의 목록 등 기본적인 질문을 물어봅니다.

스마트호스트가 메일 보내기 이 시나리오에서는 다른 컴퓨터를 통해 메일을 보냅니다. 그 다른 컴퓨터를 "스마트호스트"라고 하고, 메일을 목적지에 보내는 역할을 담당합니다. 스마트호스트는 보통 받은 메일을 저장해 놓고 있으므로, 여러분의 컴퓨터를 계속 연결해 놓지 않아도 됩니다. 메일을 받을 때는 fetchmail과 같은 프로그램으로 가져올 수도 있습니다.

보통 스마트호스트는 인터넷 서비스 회사의 메일 서버를 말합니다. 전화 접속 사용자의 경우에는 분명히 인터넷 서비스 회사의 서버를 말합니다. 스마트호스트는 회사의 메일 서버가 될 수도 있고, 내부 네트워크에 있는 다른 시스템이 될 수도 있습니다.

스마트호스트가 메일을 보내고, 로컬 메일 없음 이 옵션은 앞의 옵션과 같지만 로컬 전자메일 도메인에 대한 메일을 처리하지 않습니다. 시스템 내부의 메일은(예를 들어 시스템 관리자에게 보내는 메일은) 계속 처리합니다.

로컬 배달 시스템이 기본값으로 이렇게 설정되어 있습니다.

[4] 물론 exim4를 지우고 다른 MTA/MDA로 바꿔 버릴 수도 있습니다.

지금 설정 안함 무슨 일을 하는 지 확실히 알고 있는 경우에만 선택합니다. 이 옵션을 사용하면 메일 시스템을 설정되지 않은 상태로 남겨 놓습니다. 설정하지 않으면 메일을 보내거나 받을 수 없을 뿐만아니라 시스템 유틸리티가 보내는 중요한 메시지를 놓칠 수 있습니다.

이 시나리오가 모두 맞지 않거나, 아니면 좀 더 세밀한 설정을 하고 싶다면, 설치를 다 마친 다음에 `/etc/exim4` 디렉터리 아래의 설정 파일을 설정할 수 있습니다. `exim4`에 대한 좀 더 자세한 정보는 `/usr/share/doc/exim4`에 들어 있습니다. `README.Debian.gz` 파일에 `exim4`에 설정에 대한 더 자세한 정보가 들어 있고 어디에서 더 자세한 문서를 찾을 수 있는 지 쓰여 있습니다.

공식적인 도메인 이름이 없을 때 인터넷으로 직접 메일을 보내면 받는 서버의 스팸 방지 장치때문에 메일이 거부될 수도 있습니다. ISP의 메일 서버를 사용하는 게 좋습니다. 정말로 메일을 직접 보내고 싶다면, 기본으로 만드는 전자메일 주소가 아닌 다른 주소를 사용하십시오. `/etc/email-addresses`에 한 줄 추가하면 전자메일 주소를 바꿀 수 있습니다.

8.5 새 커널 컴파일하기

왜 커널을 직접 컴파일하고 싶어할까요? 커널 컴파일은 데비안에 탑재된 기본 커널이 모든 설정을 처리할 수 있게 된 이후로는 아마도 가장 필요 없는 일일 것입니다.

그래도 직접 커널을 컴파일하고 싶다면, 물론 가능하고 “make deb-pkg” 타겟 사용을 추천합니다. 더 많은 정보를 보려면 Debian Linux Kernel Handbook을 읽어 보십시오.

8.6 손상된 시스템 복구하기

정성을 기울여 설치한 시스템에 문제가 발생해서, 부팅하지 않을 수 있습니다. 무언가 바꾸다가 부트로더 설정이 망가졌을 수도 있고, 새로 설치한 커널이 부팅하지 않을 수도 있고, 디스크가 방사선에 맞아서 `/sbin/init` 파일의 일부를 바꾸어 놨을 수도 있습니다. 어떤 원인이든지, 이 문제를 바로잡으려면, 바로잡는 동안 작업할 시스템이 하나 필요하고, 응급복구 모드를 사용하는 게 좋습니다.

응급복구 모드로 들어가려면, 부팅 메뉴에서 **rescue**를 선택하거나, `boot:` 프롬프트에서 **rescue**라고 입력하거나 부팅 파라미터로 **rescue/enable=true** 부팅 파라미터로 부팅하십시오. 설치 프로그램의 맨 처음 화면이 나타나고, 지금 모드가 설치 모드가 아니라 응급복구 모드라는 사실을 알리는 말이 나타납니다. 너무 걱정하지 마십시오. 시스템을 엎어쓰지 않습니다! 설치 프로그램에 들어 있는 하드웨어 검색 기능을 디스크, 네트워크 장치 등을 복구하는 용도로 이용하는 것 뿐입니다.

파티션 도구 대신에 시스템의 파티션 목록이 나타나고, 그 중에 하나를 선택합니다. 보통 복구를 하려는 루트 파일시스템이 들어 있는 파티션을 선택합니다. 디스크에 있는 파티션은 물론 RAID나 LVM장치에 있는 파티션을 선택할 수도 있습니다.

가능하다면, 설치 프로그램은 선택한 파일 시스템에서 필요한 복구를 수행하려면 쉘 프롬프트를 제공하도록되어 있습니다.

선택한 루트 파일시스템에서 쉘을 실행할 수 없는 경우(예를 들어 파일 시스템이 망가진 경우), 경고 메세지를 표시하고 설치 환경 안에서 쉘을 실행합니다. 이 환경에서는 그리 많은 프로그램을 쓸 수는 없지만, 시스템을 복구하는 용도로는 충분합니다. 선택한 루트 파일시스템은 `/target` 디렉터리에 마운트되어 있습니다.

어떤 경우이든, 쉘을 빠져나가면 시스템이 다시 시작합니다.

망가진 시스템을 복구하는 일은 매우 어려울 수도 있습니다. 그리고 이 안내서는 무언가 망가졌을 때 그걸 어떻게 고치는 지에 대한 모든 사항을 설명하지 않습니다. 문제에 부딪힌 경우, 전문가에게 문의하십시오.

부록 A

설치 방법

이 문서는 debian-installer를 사용해 32-bit hard-float ARMv7용 데비안 GNU/리눅스 bullseye("armhf") 배포판을 설치하는 방법을 설명합니다. 간단히 설치 절차만 설명한 문서로, 설치할 때 필요한 정보 대부분이 들어 있습니다. 더 많은 정보가 필요한 경우 더 자세히 설명한 이 문서의 다른 부분으로 링크되어 있습니다.

A.1 들어가기 전에

설치할 때 버그를 발견하면, 5.4.5절 부분의 방법을 이용해 버그를 알려 주십시오. 이 문서로 알 수 없는 궁금한 사항이 있으면 debian-boot 메일링 리스트에(debian-boot@lists.debian.org) 알리거나 IRC에(OFTC 네트워크의 #debian-boot 채널) 질문해주십시오.

A.2 설치 프로그램 시작하기

debian-cd 팀에서 만든 설치 이미지는 데비안 CD/DVD 페이지에 있습니다. 설치 이미지를 구할 수 있는 곳은, 4.1절 부분을 참고하십시오.

일부 설치 방법에는 광학 미디어 이미지 외에 다른 이미지가 필요합니다. 4.2.1절는 데비안 미러에서 이미지를 찾는 방법에 대해 설명하고 있습니다.

다음 섹션에서는 설치 방법에 따라 어떤 이미지를 받아야 하는 지 자세히 설명합니다.

A.2.1 광학 디스크

netinst CD 이미지는 bullseye 버전을 debian-installer를 이용해 설치하는 데 많이 사용하는 설치 이미지입니다. 이 설치 방식은 이미지에서 부팅한 다음 나머지 패키지를 네트워크로 설치합니다. (그래서 이름이 "netinst"입니다.) 이 이미지에는 설치 프로그램을 실행하고 최소한의 bullseye 시스템을 구성하는데 필요한 기본 패키지가 들어 있습니다. 필요하면 설치할 때 네트워크가 필요없는 전체 CD/DVD 이미지를 받을 수도 있습니다. 설치할 때는 전체 세트에서 첫번째 이미지만 필요합니다.

마음에 드는 이미지를 다운로드하고 광학 디스크를 굽습니다.

A.2.2 네트워크 부팅

네트워크로 debian-installer를 부팅하는 것도 가능합니다. 아키텍처와 netboot 설정에 따라 다양한 방법의 네트워크 부팅 방법이 있습니다. netboot/에 들어 있는 파일을 이용해 debian-installer를 네트워크

부팅합니다.

A.2.3 하드 디스크 부팅

이동식 매체를 사용하지 않고 기존의 하드 디스크를 (거기에 다른 OS가있어도 상관 없습니다) 사용하여 설치 프로그램을 시작할 수 있습니다. `hd-media/initrd.gz`, `hd-media/vmlinuz` 및 데비안 CD/DVD 이미지를 하드 디스크의 최상위 디렉터리에 다운로드하십시오. 이미지의 파일 이름이 `.iso`로 끝나도록 하십시오. 그러면 남은 일은 initrd에서 리눅스를 부팅하는 문제입니다.

A.3 설치

설치 프로그램을 시작하면 초기 화면이 나타납니다. **Enter**를 누를 수도 있고, 아니면 다른 부팅 방법이나 파라미터 설명을 읽어 보십시오. (5.3절 참고.)

잠시 후에 언어를 선택합니다. 화살표 키로 언어를 선택하고 **Enter**를 눌러 계속 진행합니다. 그 다음에 해당 언어를 사용하는 국가 중에서 자기 국가를 선택하는 부분이 나타납니다. 여기 나오는 짧은 리스트에 자기 국가가 없다면, 세계의 모든 국가 목록에서 선택할 수도 있습니다.

키보드 설정을 확인합니다. 잘 모르겠으면 기본값을 선택하십시오.

이제 데비안 설치 프로그램이 하드웨어를 검사하고, 설치 이미지에서 나머지 부분을 읽어들이는 동안 기다리십시오.

그 다음에 설치 프로그램은 네트워크 하드웨어를 검사하고 DHCP를 통해 네트워크 설정을 합니다. 네트워크에 연결하지 않았거나 DHCP가 없다면, 네트워크를 수동으로 설정할 수 있습니다.

네트워크를 설정한 다음 사용자 계정을 만듭니다. 먼저 "루트"(관리자) 계정의 암호를 입력하고 일반 사용자 계정 생성에 필요한 정보를 입력합니다. "루트" 사용자의 암호를 지정하지 않으면 이 계정을 사용할 수 없게 되지만, 나중에 새로 설치한 시스템에 **sudo** 패키지를 설치해 관리 작업을 수행할 수 있습니다. 기본값으로 시스템에서 맨 먼저 만든 사용자는 **sudo** 명령을 사용해 root가 될 수 있도록 허가됩니다.

다음 단계는 시계 및 표준 시간대 설정입니다. 설치 프로그램이 인터넷의 타임 서버에 연결해서 시계를 올바르게 맞춥니다. 시간대는 앞에서 선택한 국가에 따라 결정합니다. 한 국가에 여러 개의 시간대가 있는 경우에만 시간대를 물어봅니다.

디스크를 파티션할 차례입니다. 먼저 전체 디스크나 디스크의 남은 공간을 자동으로 파티션 나누기 할 수 있습니다(6.3.4.2절 참고). 이 방법은 처음 설치하는 사용자나 급한 사용자에게 추천합니다. 자동 파티션을 사용하지 않으려면 메뉴에서 수동으로를 선택하십시오.

다음 화면에 파티션 테이블, 파티션의 포맷, 마운트 위치가 나타납니다. 수정하거나 지울 파티션을 선택하십시오. 자동 파티션을 할 경우, 메뉴에서 파티션 나누기를 마치고 바뀐 사항을 디스크에 쓰기를 선택하면 있는 그대로 사용합니다. 반드시 파티션 한 개를 스왑 공간으로 배정하고, 또 한 파티션을 /에 마운트하십시오. 파티션 나누기 사용하는 방법을 보려면 6.3.4절 부분을 보십시오. 부록 C에서 파티션 나누기에 대해 더 자세히 설명합니다.

파티션을 포맷하고 베이스 시스템 설치를 시작합니다. 약간의 시간이 걸릴 수 있습니다. 다 끝나면 커널을 설치합니다.

앞에서 설치한 베이스 시스템은 완전히 동작하지만 최소한의 시스템입니다. 이 시스템을 좀 더 쓸모 있게 만드려면 다음 단계에서 태스크를 선택해 패키지를 추가로 설치합니다. 패키지를 설치하기 전에 `apt`를 설정해서 어디서 패키지를 가져올 지 지정합니다. "표준 시스템" 태스크가 기본으로 선택되어 있고 보통 이 태스크는 설치해야 합니다. 그래픽 데스크톱을 사용하려면 "데스크톱 환경" 태스크를 선택하십시오. 이 단계에 관해 6.3.6.2절 부분을 참고하십시오.

마지막 단계는 부트로더 설치입니다. 컴퓨터에서 다른 운영체제를 찾으면, 그 운영체제를 부팅 메뉴에 추가하고 알려줍니다.

이제 debian-installer에서 설치 과정이 끝났다고 표시합니다. CD-ROM 또는 기타 부팅 미디어를 꺼내고 **Enter**를 눌러 다시 부팅하십시오. 그러면 새로 설치한 시스템으로 로그인할 수 있습니다. 7장에 설명되어 있습니다.

설치 과정에 대해 더 알고 싶으시면, 6장 부분을 보십시오.

A.4 설치 보고서를 보내주십시오

debian-installer를 이용해 설치에 성공했다면, 시간을 내서 설치 보고서를 보내 주십시오. 보고서를 제출하려면 가장 간단한 방법으로, reportbug 패키지를 설치하시고 (**apt install reportbug**), reportbug를 8.4.2절에 설명한 대로 설정하고, **report installation-reports**를 실행하십시오.

설치를 마치지 못했다면 데비안 설치 프로그램의 버그 때문으로 예상됩니다. 설치 프로그램을 개선하려면 개발자에게 알려야 하므로, 시간을 내서 알려 주십시오. 문제를 보고할 때 설치 보고서를 사용할 수 있습니다. 설치가 완전히 실패한다면 5.4.4절 부분을 참고하십시오.

A.5 그리고 마지막으로...

데비안 설치가 즐겁고, 데비안의 좋은 점을 느끼셨길 바랍니다. 8장을 읽는 것이 좋습니다.

부록 B

미리 설정을 이용한 설치 자동화

이 부록에서는 debian-installer의 질문에 대한 대답을 미리 설정해서 설치를 자동화하는 방법을 설명합니다.

이 부록에서 사용한 설정은 https://www.debian.org/releases/bullseye/example-preseed.txt에 들어 있는 예제 파일에서도 구할 수 있습니다.

B.1 소개

미리 설정을 하면 설치 프로그램이 실행하는 동안 나오는 질문에 대한 답을 수동으로 입력할 필요없이 미리 설정해 놓을 수 있습니다. 이렇게 하면 대부분 경우의 설치를 완전히 자동화할 수 있고, 보통 설치할 때는 쓸 수 없는 기능을 사용할 수도 있습니다.

미리 설정은 꼭 필요하진 않습니다. 미리 설정 파일을 비워 놓으면, 설치 프로그램은 일반 수동 설치와 똑같은 방식으로 동작합니다. 질문을 미리 설정해 놓으면 그 기준에 따라 설치가 다르게 동작합니다.

B.1.1 미리 설정 방법

미리 설정에 사용할 수 있는 세 가지 방법이 있습니다: initrd, 파일 그리고 네트워크입니다. initrd 미리 설정은 어떤 설치 방법과 사용해도 동작하고 더 많은 부분을 미리 설정할 수 있지만, 가장 많이 준비해야 합니다.

다음은 어떤 미리 설정을 어떤 설치 방법에서 사용할 수 있는 지 나타낸 표입니다.

설치 방법	initrd	file	network
CD/DVD/USB	예	예	예[1]
netboot	예	아니오	예
hd-media	예	예	yes[1]

l미리 설정 방법 여러 가지 사이의 중요한 차이점은, 바로 미리 설정 파일을 읽어들이고 처리할 시점입니다. initrd 미리 설정의 경우 이 시점은 설치를 시작하는 시점으로, 맨 처음으로 질문하기도 전의 시점입니다. 그 다음에 커널 명령행에 쓴 미리 설정이 시작합니다. 그러므로 커널 명령행을 (부트로더 설정이든 부트로더에서 부팅할 때 수동으로 입력하든) 편집하면 initrd에 있는 설정보다 우선하게 만들 수 있습니다. 파일 미리 설정의 경우 설치 이미지를 읽어들인 다음입니다. 네트워크 미리 설정의 경우 네트워크를 설정한 다음입니다.

[1] 하지만 네트워크에 연결된 경우에만 그렇게 하고, preseed/url을 적절히 설정합니다

중요

당연히 미리 설정 파일을 읽어들이기도 전에 처리하는 질문은 미리 설정할 수 없습니다. (여기에는 중간 혹은 낮은 우선순위에서만 표시되는 질문까지 포함합니다. 예를 들어 첫번째 하드웨어 검색이 그렇습니다.) 약간 불편하지만 이 질문을 피하는 방법은 부팅 파라미터를 통해 전달하는 방법입니다. B.2.2절 부분에서 설명합니다.

미리 설정 기능이 시작하기 전에 나타나는 질문을 간단히 피하려면, 설치 프로그램을 "자동" 모드로 시작하면 됩니다. 이렇게 하면 미리 설정하기 전에 물어보는 질문을 네트워크 연결한 다음으로 늦춰서 미리 설정이 가능합니다. 자동 모드에서는 설치를 필수 우선순위로 진행해서 중요하지 않은 많은 질문을 건너 뜁니다. 자세한 정보는 B.2.3절 부분을 참고하십시오.

B.1.2 한계

debian-installer에서 사용하는 대부분의 질문을 이 방법으로 미리 설정할 수 있지만, 몇 가지 알아둬야 할 예외가 있습니다. 전체 디스크를 다시 파티션하거나, 디스크의 빈 공간을 사용해야 합니다. 기존의 파티션을 이용할 수 없습니다.

B.2 미리 설정 사용하기

먼저 미리 설정 파일을 만들고, 그 파일을 사용하려는 위치에 놓아야 합니다. 미리 설정 파일 만들기는 이 부록의 뒤부분에서 설명합니다. 네트워크 미리 설정이나 플로피 혹은 USB 메모리에서 파일을 읽어들이는 경우, 미리 설정 파일의 위치는 아주 뻔합니다. 설치 ISO 이미지에 파일을 포함하려면 ISO 이미지를 다시 만들어야 합니다. 미리 설정 파일을 initrd에 포함하는 일은 이 문서의 범위를 벗어납니다. debian-installer 개발자 문서를 참고하십시오.

미리 설정 파일을 만들 때 기초로 사용할 수 있는 예제 파일이 https://www.debian.org/releases/bullseye/example-preseed.txt에 있습니다. 이 파일은 이 부록에 포함된 설정을 이용해서 만들었습니다.

B.2.1 미리 설정 파일 읽어들이기

initrd 미리 설정을 사용하려면, preseed.cfg 파일을 initrd의 루트 디렉터리에 놓기만 하면 됩니다. 자동으로설치 프로그램이 이 파일이 있는 지 검사한 다음 읽어들입니다.

다른 preseed 방법은 부팅할 때 어떤 파일을 읽어 들일 지 설치 프로그램에 지정해야합니다. 일반적으로 커널 부팅 파라미터로 전달합니다. 부팅할 때 수동으로 넣거나 부트 로더 설정 파일 (예 : syslinux.cfg) 편집해서 커널의 append 줄의 끝에 매개 변수를 추가합니다.

부트 로더 설정에서 미리 설정 파일을 지정하는 경우 설정을 변경하면 설치를 시작할 때 ENTER를 누를 필요가 없습니다. syslinux에서는 이 설정을하는데, syslinux.cfg에서 시간을 1로 합니다. 부트로더 설정에서 미리 설정 파일을 지정하는 경우, 설정을 바꾸면 부팅할 때 Enter를 누를 필요도 없게 할 수 있습니다. syslinux의 경우 syslinux.cfg 파일에서 timeout을 1로 하면 됩니다.

설치 프로그램이 올바른 미리 설정 파일을 읽어들이도록, 파일의 체크섬을 지정할 수도 있습니다. 현재는 md5sum만 사용 가능하고, 미리 설정 파일을 읽어들였을 때 지정한 md5sum과 체크섬이 맞지 않으면 그

파일을 사용하지 않습니다.

```
지정해야 하는 부팅 파라미터:
- 네트워크 부팅의 경우:
  preseed/url=http://호스트/파일에/대한/경로/preseed.cfg
  preseed/url/checksum=5da499872becccfeda2c4872f9171c3d
- 아니면
  preseed/url=tftp://호스트/파일에/대한/경로/preseed.cfg
  preseed/url/checksum=5da499872becccfeda2c4872f9171c3d

- 다시 만든 이미지로 부팅하는 경우:
  preseed/file=/cdrom/preseed.cfg
  preseed/file/checksum=5da499872becccfeda2c4872f9171c3d

- USB 미디어에서 설치하는 경우(미리 설정 파일을 USB 메모리의 맨 위
  디렉터리에 넣으십시오)
  preseed/file=/hd-media/preseed.cfg
  preseed/file/checksum=5da499872becccfeda2c4872f9171c3d
```

`preseed/url`은 간단히 `url`로, `preseed/file`은 `file`로, `preseed/file/checksum`은 `preseed-md5`로 줄여서 부팅 파라미터로 쓸 수 있습니다.

B.2.2 부팅 파라미터로 미리 설정하기

일부 단계에서는 미리 설정 파일을 사용할 수 없는 경우에도, 설치를 완전히 자동화할 수 있습니다. 설치 프로그램이 부팅할 때 미리 설정할 값을 하나하나 파라미터로 넘길 수 있습니다.

미리 설정 기능을 사용하려는 게 아니더라도, 특정 질문에 대한 답을 지정하고 싶으면 부팅 파라미터를 이용할 수 있습니다. 이 안내서의 다른 곳에 부팅 파라미터가 유용한 예제가 몇 가지 있습니다.

`debian-installer`에서 사용할 값을 설정하려면, 이 부록의 예제에 들어 있는 미리 설정 변수에 대해 **변수에/대한/경로=값** 형식으로 넘깁니다. 설치할 시스템의 패키지를 설정하는 데 어떤 값을 사용한다면, 그 변수의 소유자[2]를 **소유자:변수에/대한/경로=값** 형식으로 씁니다. 소유자를 지정하지 않으면 해당 변수의 값은 설치한 시스템의 debconf 데이터베이스에는 들어가지 않으므로 해당 패키지의 설정에 사용하지 않게 됩니다.

이런 식으로 질문을 미리 설정하면, 그 질문을 하지 않습니다. 질문에 대해 특정 기본값을 지정하면서, 질문을 하게 만들고 싶으면, 연산자에 "="이 아니라 "?="라고 쓰십시오. B.5.2절 부분도 참고하십시오.

부팅 파라미터에서 자주 사용하는 변수 몇개는 짧게 쓸 수 있습니다. 이 부록에 있는 예제에서는 그렇게 짧은 형식이 있으면 완전한 변수명을 쓰지 않고 짧은 형식을 사용합니다. 예를 들어 `preseed/url` 변수는 예제에서 `url`로 씁니다. 또 예제에서 `tasks`라고 쓰면 `tasksel:tasksel/first`에 해당합니다.

부팅 옵션의 "---"는 특별한 의미가 있습니다. 마지막 "---" 뒤에 오는 커널 파라미터는 설치한 시스템의 부트로더 설정으로 복사되어 들어갑니다. (설치 프로그램에서 설치하는 부트로더가 지원하는 경우.) 설치 프로그램은(미리 설정 옵션과 마찬가지로) 설치 프로그램에서 인식하는 옵션만 모두 걸러냅니다.

[2] 어떤 debconf 변수(혹은 서식)의 소유자는 보통 그 debconf 서식이 들어 있는 패키지의 이름을 말합니다. 서식과 변수는 소유자가 여러 개일 수도 있습니다. 그래서 패키지의 설정까지 지워버릴 때 소유자 정보를 이용해 해당 서식과 변수까지 지워질지 여부가 결정됩니다.

참고

현재 리눅스 커널은(2.6.9 이후) 최대 32개까지의 명령행 파라미터와 32개까지의 환경 파라미터만 쓸 수 있습니다. (설치 프로그램에서 기본으로 추가하는 파라미터 포함해서 32개입니다.) 이보다 많으면 커널이 멎어 버립니다. (이보다 오래된 버전의 커널에서는 파라미터 개수 제한이 이보다 더 작습니다.)

대부분 설치할 때 (vga=normal같은) 기본 옵션 중에 안 써도 되는 옵션이 있습니다. 그러면 미리 설정 옵션을 몇 개 더 쓸 수 있습니다.

참고

부팅 파라미터에서는 공백이 들어간 값을 지정할 수 없습니다. 따옴표로 묶어도 할 수 없습니다.

B.2.3 자동 모드

부팅 프롬프트에서 매우 간단한 명령으로 임의의 복잡한 자동 설치를 할 수 있는 데비안 설치 프로그램의 기능이 있습니다.

이 기능은 Automated install 부팅 메뉴를 선택해서 시작합니다. 일부 아키텍쳐나 일부 부팅 방식에서는 auto라고 하기도 합니다. 여기서는 auto가 파라미터가 아니라, 부팅 메뉴의 선택을 말하고, 부팅 프롬프트에서 부팅 파라미터로 추가하는 걸 말합니다.

다음은 부팅 프롬프트에서 사용할 수 있는 예입니다.

```
auto url=autoserver
```

이것은 DNS에서 autoserver 이름을 확인할 수 있고(DHCP에서 로컬 도메인이 있으면 뒤에 붙이고), 그 시스템이 DHCP 서버임을 가정합니다. example.com이라는 도메인의 사이트가 DHCP를 설정하면, http://autoserver.example.com/d-i/bullseye/./preseed.cfg에서 preseed 파일을 가져옵니다.

URL의 뒤쪽 부분은(d-i/bullseye/./preseed.cfg) auto-install/defaultroot에서 가져옵니다. 기본값으로 여기에는 bullseye 디렉터리가 들어 있습니다. 다음 버전에서는 그 버전에 해당되는 코드네임을 이 값으로 사용할 예정이고, 그러면 사람이 정해진 방법을 통해 다음 버전으로 업그레이드할 수 있습니다. /./ 부분은 맨 위 디렉터리를 가리키는 것으로, 지정한 해당 경로에(preseed/include 및 preseed/run에서 사용) 대한 상대값입니다. 이를 이용해 파일을 완전한 URL로 지정할 수도 있고, /로 시작하는 경로로 지정할 수도 있고, 마지막 미리 지정 파일이 있던 위치에 대한 상대 경로로 지정할 수도 있습니다. 이 점을 이용하면 전체 스크립트를 완전히 새로운 위치에 옮겨도 문제가 없는 포터블한 스크립트를 구성할 수 있습니다. 예를 들어 웹서버에 있던 파일을 USB 메모리에 옮겨도 문제가 없게 됩니다. 이 예제의 미리 지정 파일에서는 preseed/run 값을 /scripts/late_command.sh라고 지정하면 http://autoserver.example.com/d-i/bullseye/./scripts/late_command.s
서 파일을 가져옵니다.

로컬 네트워크에 DHCP 혹은 DNS 서버 따위가 없거나 preseed.cfg에 대한 기본값 경로를 바꾸고 싶으면, URL을 직접 사용할 수도 있습니다. /./ 부분을 사용하지 않으면 경로의 첫 부분에 대한(즉 URL에서 세번째 /) 경로가 됩니다. 다음은 로컬 네트워크에서 최소한의 지원이 있어야 동작하는 예제입니다:

```
auto url=http://192.168.1.2/파일에/대한/경로/mypreseed.file
```

위와 같이 하면 다음과 같이 동작합니다:

- URL의 프로토콜 부분을 생략하면 http라고 가정합니다.
- 호스트 이름에 점이 없으면, DHCP에서 넘겨준 도메인을 뒤에 붙입니다.
- 호스트 이름 뒤에 /가 없으면 기본 경로를 뒤에 붙입니다.

URL을 지정하는 것 외에, debian-installer의 동작과 직접 관계없지만 미리 설정 파일의 preseed/run에서 지정한 스크립트로 넘길 사항을 지정할 수도 있습니다. 현재 관련 예제는 auto-install/classes로 classes라고 줄여 쓸 수 있습니다. 다음과 같이 사용합니다:

```
auto url=example.com classes=클래스_A;클래스_B
```

클래스는 설치하려는 시스템의 종류를 지정하거나, 지역화를 지정합니다.

이 개념을 확장할 수도 있고, 확장하는 경우 auto-install 네임스페이스를 사용하는 게 보기 좋습니다. 즉 auto-install/style과 같이 스크립트에서 사용할 수 있습니다. 이렇게 해야 겠다고 생각이 들면, debian-boot@lists.debian.org 메일링 리스트에 알려 주십시오. 그래야 네임스페이스 충돌을 피하고, 여러분의 파라미터에 해당하는 줄임말을 추가할 수도 있습니다.

auto 부팅 레이블이 모든 아키텍처에서 정의된 것은 아닙니다. 커널 명령행에 파라미터 두 개, auto=true priority=critical이라고 추가하기만 하면 같은 효과를 거둘 수 있습니다. auto 커널 파라미터는 auto-install/enable의 줄임말이고 미리 설정할 수 있도록 로캘 및 키보드 질문을 뒤로 늦춥니다. 또 priority는 debconf/priority의 줄임말이고 critical로 설정하면 필수 우선순위보다 낮은 우선순위의 질문을 하지 않게 됩니다.

그 외에 DHCP를 사용할 때 설치를 자동화하면서 관심 가질만한 옵션은 다음과 같습니다: interface=auto netcfg/dhcp_timeout=60이라고 하면 처음 찾은 네트워크 인터페이스를 이용하고 DHCP 요청에 대한 응답을 좀 더 오래 기다립니다.

작은 정보

예제 스크립트와 클래스 등, 이 프레임워크를 사용하는 다양한 예제가 개발자의 웹사이트에 있습니다. 이 사이트에 있는 예제는 미리 설정을 통해 기발하고 다양한 멋진 기능을 수행하는 예제도 있습니다.

B.2.4 미리 설정할 때 쓸모 있는 줄임말

미리 설정을 사용할 경우(자동 모드) 다음 줄임말을 사용하면 좋습니다. 다음 줄임말은 질문 이름에 대한 줄임말일 뿐이고, 물론 값도 같이 쓸 수 있습니다. 예를 들어 auto=true 혹은 interface=eth0 처럼 쓸 수 있습니다.

priority	debconf/priority
fb	debian-installer/framebuffer
language	debian-installer/language
country	debian-installer/country
locale	debian-installer/locale

theme	debian-installer/theme
auto	auto-install/enable
classes	auto-install/classes
file	preseed/file
url	preseed/url
domain	netcfg/get_domain
hostname	netcfg/get_hostname
interface	netcfg/choose_interface
protocol	mirror/protocol
suite	mirror/suite
modules	anna/choose_modules
recommends	base-installer/install-recommends
tasks	tasksel:tasksel/first
desktop	tasksel:tasksel/desktop
dmraid	disk-detect/dmraid/enable
keymap	keyboard-configuration/xkb-keymap
preseed-md5	preseed/file/checksum

B.2.5 부팅 프롬프트 미리 설정 예제

다음은 부팅 프롬프트의 모양을 바꾸는 예제입니다. (필요에 따라 바꿔야 할 수 도 있습니다.)

```
# 프랑스어를 언어, 프랑스를 국가로 설정하기:
/install.amd/vmlinuz vga=788 initrd=/install.amd/gtk/initrd.gz language=fr  ↩
    country=FR --- quiet
# 영어를 언어로, 독일을 국가로 설정하고, 독일 키보드 배치를 사용:
/install.amd/vmlinuz vga=788 initrd=/install.amd/gtk/initrd.gz language=en  ↩
    country=DE locale=en_US.UTF-8 keymap=de --- quiet
# MATE 데스크톱 설치하기:
/install.amd/vmlinuz vga=788 initrd=/install.amd/gtk/initrd.gz desktop=mate- ↩
    desktop --- quiet
# web-server 태스크 설치하기:
/install.amd/vmlinuz initrd=/install.amd/initrd.gz tasksel:tasksel/first=web- ↩
    server ---
```

B.2.6 미리 설정 파일을 지정하는 데 DHCP 서버 사용하기

미리 설정 파일을 네트워크에서 다운로드하도록 지정하려면 DHCP를 사용할 수 있습니다. DHCP에서 파일 이름을 지정할 수 있습니다. 보통 이것은 netboot 파일이지만, URL 형식으로되어 있으면 네트워크 preseed을 지원하는 설치 방식은 그 URL에서 파일을 내려받은 다음 설정 파일로 사용합니다. 다음은 ISC DHCP 버전 3 서버에 dhcpd.conf 설정하는 예제입니다(데비안의 isc-dhcp-server 패키지).

```
if substring (option vendor-class-identifier, 0, 3) = "d-i" {
    filename "http://host/preseed.cfg";
```

```
}
```

위의 예에서 자신을 “d-i”라고 주장하는 DHCP 클라이언트에만 이 파일 이름을 전달하므로 일반 DHCP 클라이언트에는 아무런 영향이 없습니다. 특정 호스트에 대해서만 설정하면 네트워크의 모든 시스템을 미리 설정하지 않게 만들 수 있습니다.

DHCP preseed을 사용하는 좋은 방법은 자신의 네트워크에는 데비안 미러 같은 preseed 값만 지정하는 것입니다. 자신의 네트워크에이 방법으로 설치하면 선택한 적당한 미러를 자동으로 가져 오지만 나머지 설치 과정은 직접 지정할 수 있습니다. DHCP preseed를 이용한 데비안의 완전 자동 설치는 충분히 주의하여야 할 필요가 없습니다.

B.3 미리 설정 파일 만들기

미리 설정 파일은 **debconf-set-selections** 명령어에서 사용하는 형식으로 되어 있습니다. 미리 설정 파일의 일반적인 형식은 다음과 같습니다:

```
<소유자> <질문 이름> <질문 형식> <값>
```

파일은 `#_preseed_V1` 표시로 시작해야 합니다
미리 설정 파일을 작성할 때 지켜야 할 규칙이 있습니다.

- 형식과 값 사이에 한 개의 공백이나 탭을 넣으십시오. 공백이나 탭을 이보다 많이 쓰면 값에 그 문자가 들어갔다고 취급합니다.
- 백슬래시(“\”)를 이음 문자로 맨 뒤에 붙여서 한 줄을 여러 줄로 나눌 수 있습니다. 줄을 나눌 때 좋은 부분은 질문 이름 뒤부분입니다. 나쁜 부분은 형식과 값 사이입니다. 나눠진 줄이 한 줄로 합쳐질 때 앞/뒤의 공백 문자는 모두 공백 하나로 취급됩니다.
- 설치 프로그램에서만 사용하는 debconf 변수(서식)의 경우, 소유자를 “d-i”라고 해야 합니다. 설치한 시스템에서 사용할 변수를 미리 설정하려면, 해당 debconf 서식이 들어 있는 패키지의 이름을 사용해야 합니다. 소유자가 “d-i”가 아닌 변수만 설치한 시스템의 debconf 데이터베이스에 적용됩니다.
- 보통 질문을 미리 설정할 때 번역한 값이 아니라 영어로 된 올바른 값을 사용해야 합니다. 하지만 일부 질문의 경우 번역한 값을 사용해야 합니다. (예를 들어 `partman`에서.)
- 어떤 질문은 눈에 보이는 영문 텍스트가 아니라 코드를 값으로 받습니다.
- `#_preseed_V1` 표시로 시작합니다
- 주석은 해시 문자로 시작하는 줄이고, 그 줄이 끝날 때까지 주석입니다.

미리 설정 파일을 만드려면, B.4절에 들어 있는 예제 파일을 기초로 시작하는 방법이 가장 쉽습니다.

다른 방법으로 수동으로 설치하고 다시 부팅한 다음에, `debconf-utils` 패키지에서 **debconf-get-selections** 명령으로 debconf 데이터베이스 및 설치 프로그램의 cdebconf 데이터베이스를 한 파일로 만드는 방법이 있습니다:

```
$ echo ”#_preseed_V1” > 파일
$ debconf-get-selections --installer >> 파일
$ debconf-get-selections >> 파일
```

하지만, 이런 방법으로 만든 파일에는 미리 설정하면 안 되는 항목도 들어 있으므로, 보통 예제 파일에서 시작하는 방법이 더 좋습니다.

참고

이 방법은 설치가 끝났을 때 설치 프로그램의 cdebconf 데이터베이스가 설치한 시스템의 /var/log/installer/cdebconf에 들어 있다는 점을 이용한 것입니다. 하지만 이 데이터베이스에는 비밀 정보가 들어 있을 수도 있기 때문에 루트만 이 파일을 읽을 수 있게 되어 있습니다.

installation-report 패키지를 지우면 /var/log/installer 디렉터리 및 그 안의 모든 파일을 시스템에서 지웁니다.

각 질문에 대해 올바른 값이 무엇인지 알려면, 설치할 때 **nano**로 /var/lib/cdebconf 파일의 내용을 보면 됩니다. 원본 서식을 보려면 templates.dat 파일을 보고, 현재 값과 각 변수에 할당된 값을 보려면 questions.dat 파일을 보면 됩니다.

설치하기 전에 미리 설정 파일의 형식이 올바른 지 확인하려면, **debconf-set-selections -C preseed.cfg** 명령을 사용할 수 있습니다.

B.4 미리 설정 파일의 내용 (bullseye용)

이 부록에서 사용한 설정은 https://www.debian.org/releases/bullseye/example-preseed.txt에 들어 있는 예제 파일에서도 구할 수 있습니다.

이 예제 파일은 인텔 x86 아키텍처용으로 만들어졌습니다. 다른 아키텍처에서 설치하는 경우, 예제에서 어떤 부분은(예를 들어 키보드 선택이나 부트로더 설치) 해당 아키텍처에서 필요 없을 수도 있고 해당 아키텍처에 맞는 debconf 값으로 바꿔야 할 수도 있습니다.

다른 데비안 설치 프로그램 구성 요소의 작동 방법에 대한 자세한 내용은 6.3절에서 찾을 수 있습니다.

B.4.1 지역화

일반적인 설치에서는 지역화에 대한 질문을 먼저 합니다. 그러므로 이 값은 initrd나 커널 부팅 파라미터 방식으로만 미리 설정할 수 있습니다. 자동 모드는 (B.2.3절) auto-install/enable=true 설정을 포함합니다 (보통 줄여서 auto라고 쓰는). 이렇게 하면 지역화 질문을 지연하므로, 어떤 방법으로든 미리 설정을 할 수 있습니다.

로캘은 언어와 국가를 지정합니다. debian-installer에서 지원하는 언어와 국가라면 뭐든지 붙여서 사용할 수 있습니다. 해당 언어/국가 조합이 올바른 로캘이 아닌 경우 해당 언어에 대한 로캘을 자동으로 하나 선택합니다. 부팅 파라미터로 로캘을 지정하려면, **locale=ko_KR**과 같이 사용하십시오.

이 방법은 아주 사용하기 편하더라도, 그것은 언어의 모든 가능한 조합의 preseeding을 허용하지 않습니다. **en_NL**에 국가와 locale[3]. 이렇게 값이 개별적으로 preseeded 할 수 있습니다. 언어 및 국가는 부트 매개변수로 지정할 수 있습니다.

[3]Preseeding locale은 en_US.UTF-8로 예를 들자면 결과에 대한 설치 시스템의 기본 locale로 값을 개별적으로 preseeded해야 합니다.

```
# 로캘만 미리 설정하면 언어, 국가, 로캘 값이 설정됩니다.
d-i debian-installer/locale string en_US

# 유연하게 각각의 값을 미리 설정할 수도 있습니다.
#d-i debian-installer/language string en
#d-i debian-installer/country string NL
#d-i debian-installer/locale string en_GB.UTF-8
# 선택적으로 추가될 로캘을 지정합니다.
#d-i localechooser/supported-locales multiselect en_US.UTF-8, nl_NL.UTF-8
```

키보드 설정에서는(라틴 키보드가 아닌 아닌 키맵의 경우) 키맵과 키맵을(라틴 키보드가 아닌 키맵과 미국식 키맵 사이의) 전환하는 토글 키를 선택합니다. 설치할 때는 기본 키맵만 사용할 수 있습니다. 고급 키맵은 설치한 시스템에서만 **dpkg-reconfigure keyboard-configuration** 명령으로 사용할 수 있습니다.

```
# 키보드 선택.
d-i keyboard-configuration/xkb-keymap select us
# d-i keyboard-configuration/toggle select No toggling
```

키보드 설정을 건너뛰려면 keymap 값을 **skip-config**로 하면 됩니다. 그러면 커널 키맵을 사용합니다.

B.4.2 네트워크 설정

네트워크에서 미리 설정 파일을 읽어들이는 경우 네트워크 설정은 당연히 동작하지 않습니다. 하지만 광학 디스크나 USB 메모리로 부팅하는 경우에 네트워크 설정을 하면 좋습니다. 미리 설정 파일을 네트워크에서 읽어들이는 경우, 커널 부팅 파라미터로 네트워크 설정을 건너 뛸 수 있습니다.

미리 설정 파일을 네트워크에서 읽어들이기 전에 특정 인터페이스에서 네트워크 부팅하려면, **interface=eth1**처럼 부팅 파라미터를 사용하십시오.

네트워크를 통해 미리 설정을 하는 경우 (“preseed/url” 사용) 네트워크 설정을 미리 설정하는 게 보통 불가능하지만, 다음 방법을 이용해 피해갈 수 있습니다. 예를 들어 네트워크 인터페이스에 고정 주소를 부여하는 방법입니다. 다음 명령이 들어 있는 “preseed/run” 스크립트를 만들어서 미리 설정 파일을 읽어들인 후에 네트워크 설정을 다시 실행합니다:

```
kill-all-dhcp; netcfg
```

다음 debconf 변수가 네트워크 설정과 관계가 있습니다.

```
# 네트워크를 완전히 사용하지 않도록 설정합니다. 네트워크에 연결되지
# 않은 컴퓨터에 CD-ROM 설치를 하는 경우처럼 네트워크 관련 질문, 경고,
# 시간 초과 따위가 불필요한 경우에 좋습니다.
#d-i netcfg/enable boolean false

# 연결되어 있는 인터페이스를 선택합니다. 이러면 인터페이스가
# 여러 개 있는 경우 목록을 건너 뜁니다.
d-i netcfg/choose_interface select auto

# 특정 인터페이스를 선택할 경우:
```

```
#d-i netcfg/choose_interface select eth1

# 링크 검사 제한 시간을 다르게 설정할 때(기본값은 3초).
# 초 단위입니다
#d-i netcfg/link_wait_timeout string 10

# DHCP 서버가 느려서 응답을 기다리다가 시간이 초과되는 경우
# 다음 설정을 쓰면 됩니다.
#d-i netcfg/dhcp_timeout string 60
#d-i netcfg/dhcpv6_timeout string 60

# 네트워크 자동 설정이 기본값입니다.
# 네트워크 설정을 수동으로 하려면, 아래 줄의 주석을 지우고 그 아래에 있는
# 고정 네트워크 설정의 주석도 지우십시오.
#d-i netcfg/disable_autoconfig boolean true

# DHCP 서버가 있든 없든 모두 미리 설정 파일이 동작하게 만드려면, 아래
# 줄의 주석을 지우고 그 아래에 있는 고정 네트워크 설정의 주석도 지우십시오.
#d-i netcfg/dhcp_failed note
#d-i netcfg/dhcp_options select Configure network manually

# 고정 IP 네트워크 설정
#
# IPv4 예제
#d-i netcfg/get_ipaddress string 192.168.1.42
#d-i netcfg/get_netmask string 255.255.255.0
#d-i netcfg/get_gateway string 192.168.1.1
#d-i netcfg/get_nameservers string 192.168.1.1
#d-i netcfg/confirm_static boolean true
#
# IPv6 예제
#d-i netcfg/get_ipaddress string fc00::2
#d-i netcfg/get_netmask string ffff:ffff:ffff:ffff::
#d-i netcfg/get_gateway string fc00::1
#d-i netcfg/get_nameservers string fc00::1
#d-i netcfg/confirm_static boolean true

# DHCP에서 지정한 호스트 이름과 도메인 이름이 여기에서 설정한 것보다
# 우선합니다. 하지만 DHCP에서 호스트 이름과 도메인 이름이 넘어오는
# 경우라고 해도, 여기서 값을 설정해야 질문을 하지 않게 됩니다.
d-i netcfg/get_hostname string unassigned-hostname
d-i netcfg/get_domain string unassigned-domain

# DHCP 서버에서 넘긴 호스트 이름이나, IP에 대한 리버스 DNS와 무관하게
# 호스트 이름을 강제로 설정하려면 다음 줄의 주석을 지우고 수정하십시오.
#d-i netcfg/hostname string somehost
```

```
# 성가신 WEP 키 대화 상자를 사용하지 않습니다.
d-i netcfg/wireless_wep string
# 일부 DHCP 서버는 호스트이름을 암호처럼 사용합니다.
#d-i netcfg/dhcp_hostname string radish

# 네트워크 등의 하드웨어에 자유롭게 배포되지 않는 펌웨어가 필요한 경우, 물어보지
# 않고 그 펌웨어를 읽어들이도록 설정할 수 있습니다. 아니면 false로 하면
# 물어보지도 않게 할 수 있습니다.
#d-i hw-detect/load_firmware boolean true
```

netcfg/get_netmask를 미리 지정하지 않으면 **netcfg**는 자동으로 네트마스크를 지정합니다. 자동 설치에서는 이 변수를 seen으로 표시해야 합니다. 마찬가지로 netcfg/get_gateway를 지정하지 않으면 **netcfg**는 적당한 주소로 게이트웨이를 설정합니다. 특별한 경우로, netcfg/get_gateway를 "none"으로 설정하면 게이트웨이를 사용하지 않습니다.

B.4.3 네트워크 콘솔

```
# SSH을 통해 원격 설치를 하면서 network-console 컴포넌트를 사용할 경우 다음
# 설정을 사용합니다. 이후의 모든 설치를 수동으로 하는 경우에만 이렇게 합니다.
#d-i anna/choose_modules string network-console
#d-i network-console/authorized_keys_url string http://10.0.0.1/openssh-key
#d-i network-console/password password r00tme
#d-i network-console/password-again password r00tme
```

network-console 관련된 더 많은 정보는 6.3.10절에 있습니다.

B.4.4 미러 사이트 설정

사용하는 설치 방법에 따라서, 미러 사이트를 이용해 설치 프로그램의 추가 컴포넌트, 베이스 시스템을 다운로드할 수 있습니다. 또 설치를 끝낸 시스템에서 /etc/apt/sources.list 파일을 설정하는 데 미러 사이트를 이용할 수 있습니다.

mirror/suite 파라미터로 설치할 시스템의 세트를 결정합니다.

mirror/udeb/suite 파라미터는 설치 프로그램의 추가 컴포넌트의 세트를 설정합니다. 실제 구성 요소를 네트워크로 다운로드하는데 도움이됩니다. 또한 설치에 사용할 설치 방법을위한 initrd를 생성하려면이 쌍이 일치해야합니다. 일반적으로 설치 프로그램은 자동으로 올바른 값을 설정하므로 설정할 필요가 없습니다.

```
# 미러 프로토콜:
# ftp의 경우, mirror/country 문자열은 설정할 필요가 없습니다.
# 미러 프로토콜에 사용할 기본값: http
#d-i mirror/protocol string ftp
d-i mirror/country string manual
d-i mirror/http/hostname string ftp.kr.debian.org
d-i mirror/http/directory string /debian
d-i mirror/http/proxy string
```

```
# 설치할 세트
#d-i mirror/suite string testing
# 설치 프로그램을 읽어들일 세트(옵션).
#d-i mirror/udeb/suite string testing
```

B.4.5 계정 설정

루트 계정의 암호와 맨 처음 만들 일반 사용자의 이름 및 암호도 미리 설정할 수 있습니다. 암호의 경우 일반 텍스트 값을 그대로 쓸 수도 있고 crypt(3) 해시값을 쓸 수도 있습니다.

주의

미리 설정한 암호는 안전하지 않습니다. 미리 설정 파일을 읽을 수 있는 사람은 암호도 알 수 있기 때문입니다. 해시 형태로 암호를 저장하면 (다수의 암호를 입력하는데 좋은 DES 나 MD5와 같은 취약한 알고리즘을 사용하지 않는 한) 보안상 안전하다고 여겨집니다. 추천하는 암호 해시 알고리즘은 SHA-256 및 SHA512입니다.

```
# 루트 계정을 만들지 않고 넘어갑니다.  (일반 유저는 sudo를 사용할
# 수 있습니다.).
#d-i passwd/root-login boolean false
# 아니면 일반 사용자를 만들지 않고 넘어갈 수 있습니다.
#d-i passwd/make-user boolean false

# 루트 암호, 암호 원문 텍스트를 직접 쓸 수도 있고
#d-i passwd/root-password password r00tme
#d-i passwd/root-password-again password r00tme
# 아니면 crypt(3) 해시로 암호화된 암호를 쓸 수도 있습니다.
#d-i passwd/root-password-crypted password [crypt(3) hash]

# 아니면 일반 사용자 계정을 하나 만듭니다.
#d-i passwd/user-fullname string Debian User
#d-i passwd/username string debian
# 일반 사용자 암호, 암호 원문 텍스트를 직접 쓸 수도 있고
#d-i passwd/user-password password insecure
#d-i passwd/user-password-again password insecure
# 아니면 crypt(3) 해시로 암호화된 암호를 쓸 수도 있습니다.
#d-i passwd/user-password-crypted password [crypt(3) hash]
# 기본값이 아닌 지정한 UID 값으로 첫번째 사용자를 만듭니다.
#d-i passwd/user-uid string 1010

# 사용자 계정은 표준으로 정해진 그룹에 들어갑니다. 강제로
# 그룹을 지정하려면 다음과 같이 합니다.
#d-i passwd/user-default-groups string audio cdrom video
```

passwd/root-password-crypted 및 passwd/user-password-crypted 변수의 값으로 "!"를 써서 미리 설정할 수 있습니다. 이 경우 해당하는 계정을 사용할 수 없습니다. 루트 계정은 이렇게 하는 게 편리할 수도 있습니다. 물론 루트 계정을 이렇게 하면 시스템 관리를 할 수 있는 다른 방법이 있거나 루트 로그인을 할 수 있는 다른 방법이 있어야 합니다. (예를 들어 SSH 키 인증을 쓰거나 **sudo**를 사용하는 방법)

암호에 대한 SHA-512 기반 crypt(3) 해시를 만드려면 다음 명령을 사용할 수 있습니다(whois 패키지에 들어 있습니다):

```
mkpasswd -m sha-512
```

B.4.6 시계 및 시간대 설정

```
# 하드웨어 시계를 UTC로 할 지 여부를 결정합니다
d-i clock-setup/utc boolean true

# $TZ로 설정 가능한 값은 뭐든지 쓸 수 있습니다. 설정 가능한
# 값은 /usr/share/zoneinfo/ 아래의 내용을 참고하십시오.
d-i time/zone string US/Eastern

# 설치하면서 시계를 맞출 때 NTP를 사용할 지 여부를 설정합니다.
d-i clock-setup/ntp boolean true
# 사용할 NTP 서버. 보통 기본값을 사용하는 게 좋습니다.
#d-i clock-setup/ntp-server string ntp.example.com
```

B.4.7 파티션하기

하드 디스크 파티션에 preseed를 사용하는 것은 partman-auto에서 지원하는 기능에 한정되고 있습니다. 파티션은 디스크의 빈 영역과 전체 디스크 중 하나를 선택해야합니다. 디스크의 구성은 미리 정의된 요리법, 레시피 파일을 사용하여 사용자 정의 레시피 미리 설정 파일에 쓴 레시피에서 선택할 수 있습니다.

RAID, LVM 및 암호화를 사용하여 고급 파티션 설정의 Preseeding는 지원 되지만, 가능한 한 완전한 유연성이 non-preseeded 설치하는 동안 파티션을 하지 않을 경우입니다.

아래의 예제는 사용법에 대한 기본적인 정보를 제공합니다. 자세한 내용은 debian-installer 패키지에 포함되어 있는 파일 partman-auto-recipe.txt 및 partman-auto-raid-recipe.txt를 참조하십시오. 두 파일은 debian-installer 소스 저장소에도 들어 있습니다. 지원되는 기능은 릴리스마다 달라질 수 있습니다.

주의

디스크의 ID는 디스크의 드라이버를 읽어들이는 순서에 따라 다릅니다. 시스템에 디스크가 여러 개 있는 경우, 미리 설정을 이용하기 전에 올바른 디스크를 선택하도록 하십시오.

B.4.7.1 파티션 예제

```
# 시스템에 빈 공간이 있을 때 그 공간만 파티션하게 설정할 수 있습니다.
# 아래의 partman-auto/method 설정이 있을 때만 적용됩니다.
#d-i partman-auto/init_automatically_partition select biggest_free

# 다른 방법으로, 파티션할 디스크를 직접 지정할 수 있습니다. 시스템에 디스크가 1개
# 뿐인 경우 설치 프로그램에서는 그 디스크를 사용합니다. 하지만 여러 개인 경우 장치
# 이름은 전통적인, devfs 형식이 아닌 형식으로 지정합니다. (예를 들어 /dev/sda.
# /dev/discs/disc0/disc 형식이 아님.)
# 예를 들어, 첫 번째 SCSI/SATA 하드 디스크를 사용하려면:
#d-i partman-auto/disk string /dev/sda
# 추가로 사용할 파티션 방식을 지정해야 합니다.
# 현재 사용할 수 있는 파티션 방식은
# - regular: 아키텍쳐에 따라 일반적인 파티션 방식을 사용
# - lvm:     파티션에 LVM을 사용
# - crypto:  암호화한 파티션에 LVM을 사용
d-i partman-auto/method string lvm

# LVM 볼륨 그룹에 사용할 용량을 직접 지정할 수 있습니다. 그 크기를 단위와
# 함께 지정하거나 (예: 20 GB), 빈 공간의 퍼센트값을 지정하거나, 또는 'max'
# 키워드로 빈 공간 전체를 지정할 수 있습니다.
d-i partman-auto-lvm/guided_size string max

# 자동으로 파티션할 디스크 중에 과거 LVM 설정이 남아 있을 경우, 경고 메시지를
# 받게 됩니다. 이 경고를 미리 설정으로 무시할 수 있습니다...
d-i partman-lvm/device_remove_lvm boolean true
# 기존에 소프트웨어 RAID 어레이가 설정된 경우에도 마찬가지로 적용됩니다:
d-i partman-md/device_remove_md boolean true
# LVM 파티션 쓰기를 확인하는 경우도 마찬가지입니다:
d-i partman-lvm/confirm boolean true
d-i partman-lvm/confirm_nooverwrite boolean true

# 다음 세 가지 파티션 스타일 중에 하나를 사용할 수 있습니다:
# - atomic: 모든 파일을 하나의 파티션에
# - home:   /home 파티션 별도로 분리
# - multi:  /home, /var, /tmp 파티션 별도로 분리
d-i partman-auto/choose_recipe select atomic

# 아니면 직접 파티션 스타일을 지정할 수도 있습니다...
# d-i 환경에 레시피 파일이 들어 있으면, 그 파일을 지정하면 됩니다.
#d-i partman-auto/expert_recipe_file string /hd-media/recipe

# 레시피 파일이 따로 없는 경우, 전체 레시피 파일 내용을(논리적으로) 한
# 줄에 넣을 수도 있습니다. 다음 예제는 작은 /boot 파티션을 만들고,
# 적당한 스왑 파티션을 만들고, 나머지 공간을 루트 파티션에 사용합니다:
#d-i partman-auto/expert_recipe string                         \
```

```
#      boot-root ::                                             \
#              40 50 100 ext3                                   \
#                      $primary{ } $bootable{ }                 \
#                      method{ format } format{ }               \
#                      use_filesystem{ } filesystem{ ext3 }     \
#                      mountpoint{ /boot }                      \
#              .                                                \
#              500 10000 1000000000 ext3                        \
#                      method{ format } format{ }               \
#                      use_filesystem{ } filesystem{ ext3 }     \
#                      mountpoint{ / }                          \
#              .                                                \
#              64 512 300% linux-swap                           \
#                      method{ swap } format{ }                 \
#              .

# 전체 레시피 형식은 'debian-installer' 패키지나 D-I 소스 저장소의
# partman-auto-recipe.txt 파일에 쓰여 있습니다. 이 파일에는 파일 시스템
# 레이블, 볼륨 그룹 이름, 볼륨 그룹에 물리 장치 포함하기 따위를
# 설정하는 방법도 쓰여 있습니다.

## EFI를 위한 파티션 만들기
# 시스템에 EFI 파티션이 필요하면 다음과 같은 사항을 위 레시피에
# 레시피 첫번째 항목으로 추가할 수 있습니다:
#               538 538 1075 free                               \
#                      $iflabel{ gpt }                          \
#                      $reusemethod{ }                          \
#                      method{ efi }                            \
#                      format{ }                                \
#               .                                               \
#
# 위는 amd64 아키텍쳐를 위한 내용입니다. 다른 아키텍쳐에서 자세한
# 사항은 다를 수도 있습니다. D-I 소스 저장소의 'partman-auto'
# 패키지에 따라 할 수 있는 예제가 있을 수도 있습니다.

# 다음과 같이 설정하면 partman에서 확인 질문을 하지 않고 자동으로
# 파티션을 합니다. 위의 방법 중에 한 가지로 파티션 방법을 지정해야
# 합니다.
d-i partman-partitioning/confirm_write_new_label boolean true
d-i partman/choose_partition select finish
d-i partman/confirm boolean true
d-i partman/confirm_nooverwrite boolean true

# UEFI 부팅 강제 ('BIOS 호환성'이 없음). 기본값: false.
#d-i partman-efi/non_efi_system boolean true
# GPT 파티션 테이블을 사용 - EFI에 필수
```

```
#d-i partman-partitioning/choose_label select gpt
#d-i partman-partitioning/default_label string gpt

# 디스크 암호화를 사용할 때, 그 전에 파티션 내용 청소를 건너 뜁니다.
#d-i partman-auto-crypto/erase_disks boolean false
```

B.4.7.2 RAID를 사용해 파티션하기

소프트웨어 RAID 파티션을 설정하거나 preseed를 사용하여 수 있습니다. 지원은 RAID 0, 1, 5, 6, 10, 비상용 어레이 및 예비 장치를 지정합니다.

> 주의
>
> 
>
> 이런 방식의 자동 파티션은 잘못되기 쉽습니다. 또 이 기능은 debian-installer 개발자가 별로 테스트하지 않는 기능입니다. 여러가지 방식을 올바르게(규칙에 맞으면서 충돌하지 않게) 설정하는 책임은 사용자에게 있습니다. 문제가 발생하면 /var/log/syslog 파일을 확인하십시오.

```
# method 값은 "raid"로 설정합니다.
#d-i partman-auto/method string raid
# 파티션할 디스크를 지정합니다. 디스크는 모두 같은 레이아웃이어야
#하므로, 크기가 같을 경우에만 다음 설정이 동작합니다.
#d-i partman-auto/disk string /dev/sda /dev/sdb

# 그리고 사용할 물리 파티션을 지정합니다.
#d-i partman-auto/expert_recipe string \
#      multiraid ::                                          \
#              1000 5000 4000 raid                           \
#                      $primary{ } method{ raid }            \
#              .                                             \
#              64 512 300% raid                              \
#                      method{ raid }                        \
#              .                                             \
#              500 10000 1000000000 raid                     \
#                      method{ raid }                        \
#              .

# 마지막으로 이전에 지정한 파티션을 어떻게 RAID에서 사용할지
# 지정합니다. 논리 파티션에 올바른 파티션 번호를 사용하도록 하십시오.
# RAID 레벨 0, 1, 5, 6, 10을 지원합니다. 장치는 "#"문자로 구분합니다.
# 파라미터는 다음과 같습니다:
# <raidtype> <devcount> <sparecount> <fstype> <mountpoint> \
#          <devices> <sparedevices>
```

```
#d-i partman-auto-raid/recipe string \
#    1 2 0 ext3 /                    \
#          /dev/sda1#/dev/sdb1       \
#    .                               \
#    1 2 0 swap -                    \
#          /dev/sda5#/dev/sdb5       \
#    .                               \
#    0 2 0 ext3 /home                \
#          /dev/sda6#/dev/sdb6       \
#    .

# 더 자세한 정보는 'debian-installer' 패키지나 D-I 소스 저장소의
# partman-auto-raid-recipe.txt 파일에 있습니다.

# 다음 설정을 하면 partman에서 확인 질문 없이 파티션을 자동으로 진행합니다.
d-i partman-md/confirm boolean true
d-i partman-partitioning/confirm_write_new_label boolean true
d-i partman/choose_partition select finish
d-i partman/confirm boolean true
d-i partman/confirm_nooverwrite boolean true
```

B.4.7.3 파티션 마운트 방법 조정하기

파일 시스템은 장치 이름이 바뀌더라도 UUID(universally unique identifier)를 키로 해서 마운트합니다. UUID는 길어서 알아보기 어려우므로, 전통적인 장치 이름에 따라 마운트할 수도 있고, 레이블을 이용해 마운트할 수도 있습니다. 레이블에 따라 마운트할 경우, 레이블이 없는 파일 시스템은 UUID를 사용해 마운트합니다.

LVM 논리 볼륨처럼 고정된 이름의 장치는 UUID가 아니라 계속 전통적인 이름을 사용합니다.

주의

전통적인 장치 이름은 부팅할 때 장치를 발견한 순서에 따라 달라질 수 있습니다. 그래서 잘못된 파일 시스템을 마운트하는 실수를 저지를 수 있습니다. 마찬가지로 레이블도 새로운 디스크나 USB 드라이브 따위를 연결했을 때 레이블이 충돌할 수 있고 그 경우 시스템이 어떻게 동작할지 확신할 수 없습니다.

```
# 기본값은 UUID로 마운트하는 것이지만, 전통적인 장치 이름을 사용하려면
# "traditional"을 사용할 수 있고, 파일 시스템 레이블을 사용하려면
# "label"을 사용합니다. 시도가 실패하면 UUID를 사용합니다.
#d-i partman/mount_style select uuid
```

B.4.8 기본 시스템 설치

이 상태에서는 미리 설정할 수 있는 부분이 별로 많지 않습니다. 유일하게 신경 쓸 부분은 커널 설치에 관한 질문입니다.

```
# APT에서 권장 패키지를 설치하지 않도록 설정합니다. 이 옵션을 사용하면
# 불완전한 시스템이 될 수도 있으므로, 아주 경험 많은 사용자만 사용해야
# 합니다.
#d-i base-installer/install-recommends boolean false

# 설치할 커널 이미지 패키지(또는 메타 패키지). 커널을 설치하지 않으면 "none"을
# 사용합니다.
#d-i base-installer/kernel/image string linux-image-686
```

B.4.9 APT 설정

/etc/apt/sources.list의 설정과 기본 설정 옵션은 설치 방법과 그 이전의 질문에 어떻게 답했냐에 따라 완전히 자동화합니다. 추가적으로 다른 저장소를 지정할 수 있습니다.

```
# Choose, if you want to scan additional installation media
# (default: false).
d-i apt-setup/cdrom/set-first boolean false
# You can choose to install non-free and contrib software.
#d-i apt-setup/non-free boolean true
#d-i apt-setup/contrib boolean true
# Uncomment the following line, if you don't want to have the sources.list
# entry for a DVD/BD installation image active in the installed system
# (entries for netinst or CD images will be disabled anyway, regardless of
# this setting).
#d-i apt-setup/disable-cdrom-entries boolean true
# Uncomment this if you don't want to use a network mirror.
#d-i apt-setup/use_mirror boolean false
# Select which update services to use; define the mirrors to be used.
# Values shown below are the normal defaults.
#d-i apt-setup/services-select multiselect security, updates
#d-i apt-setup/security_host string security.debian.org

# Additional repositories, local[0-9] available
#d-i apt-setup/local0/repository string \
#       http://local.server/debian stable main
#d-i apt-setup/local0/comment string local server
# Enable deb-src lines
#d-i apt-setup/local0/source boolean true
# URL to the public key of the local repository; you must provide a key or
# apt will complain about the unauthenticated repository and so the
```

```
# sources.list line will be left commented out.
#d-i apt-setup/local0/key string http://local.server/key
# If the provided key file ends in ".asc" the key file needs to be an
# ASCII-armoured PGP key, if it ends in ".gpg" it needs to use the
# "GPG key public keyring" format, the "keybox database" format is
# currently not supported.

# By default the installer requires that repositories be authenticated
# using a known gpg key. This setting can be used to disable that
# authentication. Warning: Insecure, not recommended.
#d-i debian-installer/allow_unauthenticated boolean true

# Uncomment this to add multiarch configuration for i386
#d-i apt-setup/multiarch string i386
```

B.4.10 패키지 선택

태스크는 원하는 대로 설치할 수 있습니다. 현재 이 문서를 쓰는 시점에 사용할 수 있는 태스크는 다음과 같습니다:

- **standard** (표준 도구)
- **desktop** (그래픽 데스크톱)
- **gnome-desktop** (그놈 데스크톱)
- **xfce-desktop** (XFCE 데스크톱)
- **kde-desktop** (KDE 플라즈마 데스크톱)
- **cinnamon-desktop** (시나몬 데스크톱)
- **mate-desktop** (MATE 데스크톱)
- **lxde-desktop** (LXDE 데스크톱)
- **web-server** (웹 서버)
- **ssh-server** (SSH 서버)

태스크를 설치하지 않을 수도 있고, 다른 방법으로 패키지를 설치할 수 있습니다. **표준 시스템** 태스크는 항상 포함하시길 권장합니다.

tasksel 대화창을 아예 표시하지 않으려면, `pkgsel/run_tasksel` 값을 미리 설정하십시오. (이 경우 tasksel에서 아무 패키지도 설치하지 않습니다.)

태스크로 설치한 패키지 외에 패키지를 더 설치하려면, `pkgsel/include` 파라미터를 사용하면 됩니다. 이 파라미터의 값은 쉼표나 공백으로 구분할 수 있으므로, 커널 명령행에서도 쉽게 사용할 수 있습니다.

```
#tasksel tasksel/first multiselect standard, web-server, kde-desktop

# tasksel 대화창이 아예 표시되지 않게 하고 싶으면 (그리고 아무 패키지도 설치하지
# 않고 싶으면):
#d-i pkgsel/run_tasksel boolean false

# 추가로 설치할 패키지
#d-i pkgsel/include string openssh-server build-essential
# debootstrap 다음에 패키지를 업그레이드할 지 여부
# 사용 가능한 값은: none, safe-upgrade, full-upgrade
#d-i pkgsel/upgrade select none

# 원한다면 어떤 소프트웨어를 설치했는지 설치 프로그램에서 보고서를 보낼 수
# 있습니다. 보고하지 않는 게 기본값이지만, 보고서를 보내면 데비안 프로젝트에서
# 어떤 소프트웨어를 더 많이 사용하는지 판단하고, 첫번째 CD/DVD에 포함해야 할지
# 여부를 결정하는데 도움이 됩니다.
#popularity-contest popularity-contest/participate boolean false
```

B.4.11 설치 마치기

```
# 시리얼 콘솔에서 설치하면, 일반 가상 콘솔은(VT1-VT6) /etc/inittab에서
# 막습니다. 다음의 주석을 지우면 가상 콘솔을 막지 않습니다.
#d-i finish-install/keep-consoles boolean true

# 설치가 끝났다는 마지막 메시지를 표시하지 않습니다.
d-i finish-install/reboot_in_progress note

# 다음과 같이 하면 다시 시작할 때 CD를 빼지 않습니다.
# 경우에 따라서는 CD를 빼지 않는 게 좋을 수 있습니다.
#d-i cdrom-detect/eject boolean false

# 다음과 같이 하면 설치가 끝났을 때, 설치한 시스템으로
# 다시 시작하지 않고 셧다운합니다.
#d-i debian-installer/exit/halt boolean true
# 다음과 같이 하면 컴퓨터의 전원도 끕니다.
#d-i debian-installer/exit/poweroff boolean true
```

B.4.12 기타 패키지 미리 설정

```
# 어떤 소프트웨어를 설치하느냐에 따라, 혹은 설치하는 중에 무언가 잘못되는
# 경우, 다른 질문을 물어볼 수도 있습니다. 물론 이 질문도 미리 설정할 수
# 있습니다. 설치하는 동안 물어볼 수 있는 모든 질문의 목록을 받고 싶다면,
# 설치를 한 다음에 다음 명령어를 실행하십시오:
#   debconf-get-selections --installer > file
```

```
#   debconf-get-selections >> file
```

B.5 고급 옵션

B.5.1 설치할 때 임의의 명령어 실행하기

미리 설정 도구의 매우 강력하고도 유연한 옵션은, 설치 특정 시점에 명령어와 스크립트를 실행하는 기능입니다.

대상 시스템의 파일 시스템을 마운트했으면, /target 아래에 그 파일 시스템이 있습니다. 설치 CD를 사용하고 마운트하면, 그 내용은 /cdrom 아래에 있습니다.

```
# d-i 미리 설정은 원래부터 보안에 안전하지 않습니다. 설치 프로그램 중의
# 어느 부분도 버퍼 오버플로우나 그 밖의 방법으로 미리 설정 파일의 값을
# 조작하는 공격을 검사하지 않습니다. 믿을 만한 곳에 있는 미리 설정 파일만
# 사용하십시오! 설치 프로그램 안에서 어떤 쉘 명령어라도 실행할 수 있는
# 방법이 만들어져 있습니다. 위험하지만 이 방법은 매우 유용하므로,
# 다음과 같이 설치 프로그램 내에서 쉘 명령어를 실행할 수 있습니다.

# 다음 첫 번째 명령어는 미리 설정 파일을 읽어들인 직후에 가능한 빨리
# 실행합니다.
#d-i preseed/early_command string anna-install some-udeb
# 다음 명령은 파티션 프로그램이 시작하기 직전에 실행합니다. 여기에는
# 디스크의 상태에 따라 다르게 동적으로 파티션하는 미리 설정을 사용하면
# 편리합니다. (preseed/early_command 명령이 실행하는 시점에서는
# 디스크의 상태를 알 수 없습니다.)
#d-i partman/early_command \
#       string debconf-set partman-auto/disk "$(list-devices disk | head -n1)"
# 다음 명령은 설치를 끝내기 직전에 실행합니다. 그러나 /target 디렉터리는 아직
# 사용할 수 있는 시점입니다. /target 디렉터리로 chroot해서 직접 사용할
# 수 있고 패키지를 쉽게 설치하려면 apt-install과 in-target명령을 사용할
# 수 있습니다.
#d-i preseed/late_command string apt-install zsh; in-target chsh -s /bin/zsh
```

B.5.2 미리 설정을 이용해 기본값 바꾸기

미리 설정으로 질문에 대한 기본값을 바꾸면서, 그래도 그 질문을 받도록 만들 수 있습니다. 이렇게 하려면 해당 서식에 대한 값을 설정한 다음에 seen 플래그를 "false"로 놓으면 됩니다.

```
d-i foo/bar string value
d-i foo/bar seen false
```

부팅 파라미터로 preseed/interactive=true라고 설정하면 모든 질문에 대해서 같은 효과를 거둘 수 있습니다. 이 기능은 미리 설정 파일을 테스트하거나 디버깅하는 데도 좋습니다.

주의할 점이, "d-i" 소유자는 설치 프로그램에서 사용하는 변수에만 사용해야 합니다. 대상 시스템에 설치한 패키지에 관련된 변수에 대해서는 그 패키지의 이름을 사용해야 합니다. B.2.2절 부분의 각주를 보십시오.

부팅 파라미터를 이용해 미리 설정을 하는 경우, "?=" 연산자를 사용해서 해당 질문을 물어보도록 만들 수 있습니다. 예를 들어 **어쩌구/저쩌구?=값**와 같이(아니면 **소유자:어쩌구/저쩌구?=값**) 합니다.

디버깅 정보를 자세히 보려면 DEBCONF_DEBUG=5 부팅 파라미터를 사용하십시오. 이러면 debconf에서 각 변수의 현재 설정 및 각 패키지의 설치 스크립트의 진행 상태에 대해 더 자세히 표시합니다.

B.5.3 미리 설정 파일을 분리해서 사용하기

미리 설정 파일에서 다른 미리 설정 파일을 포함할 수도 있습니다. 파일에 들어 있는 설정은 앞에서 읽어들인 파일에 들어 있는 설정을 덮어 씁니다. 이 방법을 이용해서, 예를 들어 파일 하나에 일반적인 네트워크 설정을 집어 넣고 세세한 설정을 다른 파일에 집어 넣는 식으로 활용이 가능합니다.

```
# 여러 개 파일을 공백으로 구분해서 쓸 수도 있습니다. 그러면 모든
# 파일을 읽어들입니다. 물론 포함한 파일은 그 안에 preseed/include가
# 들어 있을 수 있습니다. 주의할 점으로, 파일 이름이 상대 경로인 경우 그
# 파일이 포함되어 들어가는 파일이 있는 같은 디렉터리에서 찾게 됩니다.
#d-i preseed/include string x.cfg

# 설치 프로그램은 미리 설정 파일을 사용하기 전에 그 파일의 체크섬을
# 검사합니다. 현재는 md5sum만 지원하고, md5sum을 포함하는 파일과 같은
# 순서로 쓰십시오.
#d-i preseed/include/checksum string 5da499872becccfeda2c4872f9171c3d

# 좀 더 유연하게 하려면, 다음과 같이 하면 미리 설정 파일의 이름을
# 출력하는 쉘 명령어를 출력하고, 그 파일을 포함합니다.
#d-i preseed/include_command \
#       string if [ "`hostname`" = bob ]; then echo bob.cfg; fi

# 이 중에거 가장 유연한 것으로, 프로그램을 다운로드하고 이를 실행할 수
# 있습니다. 이 프로그램은 debconf 데이터베이스를 조작하려면
# debconf-set과 같은 명령을 사용할 수 있습니다. 여러 개의 스크립트를
# 공백으로 구분해서 쓸 수도 있습니다. 파일 이름이 상대경로로 되어 있으면
# 프로그램을 실행하는 미리 설정 파일이 있는 디렉터리에서 파일을 찾습니다.
#d-i preseed/run string foo.sh
```

initrd 혹은 파일을 이용한 미리 설정 단계에서, 파일 안에 다시 preseed/url을 설정해서 네트워크 미리 설정을 겹쳐 넣을 수도 있습니다. 이렇게 하면 네트워크가 연결되었을 때 미리 설정을 읽어들이게 됩니다. 이와 같이 하는 경우에는 주의해야 합니다. 미리 설정을 실행하는 두 개의 별도의 단계가 있기 때문입니다. 예를 들어서 preseed/early 명령을 한 번 더 실행할 수 있고, 두 번째가 네트워크가 연결된 다음에 실행될 수 있습니다.

부록 C

데비안에서 파티션 나누기

C.1 데비안 파티션 및 크기 정하기

최소한의 구성으로 GNU/Linux는 자신을 위해 하나 이상의 파티션을 필요로합니다. 전체 운영 체제, 응용 프로그램, 개인 파일은 하나의 파티션에 저장됩니다. 많은 사람이 이와 swap 파티션도 필요하다고 생각하는 것 같습니다만, 이것은 엄밀하게 올바르지는 않습니다. “Swap”은 운영 체제가 가진 메모리의 임시 공간으로, 이것을 이용하면 시스템은 디스크 장치를 “가상 메모리”로 사용할 수 있게됩니다. swap을 별도의 파티션에 두면, Linux에서 이용이 훨씬 더 효율적입니다. Linux 일반적인 파일을 swap으로 사용할 수 있지만 이것은 권장하지 않습니다.

하지만 대부분의 사람이 최소한 필요한 것보다 많은 파티션을 GNU/Linux에 할당합니다. 파일 시스템을 몇 개의 작은 파티션에 나누는 이유는 2 가지가 있습니다. 첫 번째는 안전성입니다. 만약 우연히 무언가가 파일 시스템을 파괴해도 일반적으로 그 영향을받는 것은 하나의 파티션만입니다. 따라서 시스템의 일부분만(잘 보관해두고 있던 백업에서) 복구하면됩니다. 이런 이유에서 “루트 파티션”은 따로 하는 것을 고려하십시오. 여기에는 시스템의 가장 기본적인 구성 부분이 들어 있고, 만약 다른 파티션에 손상이 생기더라도, Linux를 시작하여 시스템을 바로잡을 수 있습니다. 시스템을 처음부터 다시 설치해야하는 듯한 문제를 막을 수 있습니다.

두번째 이유는 보통 업무용 컴퓨터에서 더 중요하지만, 컴퓨터를 어떻게 사용하느냐에 따라 다릅니다. 예를 들어 대량을 스팸 메일을 받는 메일 서버에서는 금방 파티션 하나가 꽉 찹니다. 그 메일 서버에서 `/var/mail`을 별도의 파티션에 만들었다면, 스팸메일을 받더라도 시스템의 다른 부분은 계속 동작합니다.

파티션 여러 개의 사용할 경우 유일한 단점은, 파티션에 필요한 크기를 미리 알기 힘들다는 점입니다. 파티션을 너무 작게 만들면 시스템을 새로 설치하거나 그 파티션에 있는 파일을 자주 다른 파티션으로 옮겨야 합니다. 반면 파티션을 너무 크게 만들면 다른 곳에서 쓸 수 있는 용량을 낭비하는 셈이 됩니다. 디스크 가격이 저렴해 졌지만 낭비할 필요는 없습니다.

C.2 디렉터리 구조

디렉터리와 파일 이름에 대해 데비안 GNU/리눅스는 Filesystem Hierarchy Standard에 따릅니다. 이 표준을 준수함으로써 사용자들과 유저 프로그램은 파일과 디렉터리의 위치를 예상하기 쉽습니다. 루트 디렉터리는 슬래쉬 `/`로 표시됩니다. 루트 수준에는 데비안 시스템은 반드시 다음과 같은 디렉터리가 포함됩니다:

디렉터리	내용

디렉터리	내용
bin	핵심 명령어 바이너리
boot	부트로더에서 필요한 고정 파일
dev	장치 파일
etc	이 호스트의 시스템 설정
home	사용자 홈 디렉터리
lib	핵심 공유 라이브러리 및 커널 모듈
media	이동식 미디어의 마운트 위치가 들어 있습니다
mnt	파일 시스템을 임시로 마운트하는 마운트 위치
proc	시스템 정보를 저장하는 가상 디렉터리
root	루트 사용자의 홈 디렉터리
run	실행할 때 바뀌는 데이터
sbin	핵심 시스템 바이너리
sys	시스템 정보를 저장하는 가상 디렉터리
tmp	임시 파일
usr	이차 디렉터리 구조
var	자주 바뀌는 데이터
srv	시스템 서비스의 데이터
opt	별도의 응용 소프트웨어 패키지

아래의 목록은 디렉터리와 파티션에 대해 고려할 사항입니다. 실제 시스템 사용량은 시스템의 설정과 사용 용도에 따라 달라집니다. 아래 권장 사항은 파티션할 때 참고만 하십시오.

- /etc, /bin, /sbin, /lib, /dev는 반드시 루트 파티션(/)에 들어 있어야 합니다. 그렇지 않으면 부팅에 문제가 발생합니다. 루트 파티션은 일반적으로 250–350MB 정도가 필요합니다.
- /usr: 모든 유저 프로그램(/usr/bin)과 라이브러리(/usr/lib)와 시스템 문서(/usr/share/doc) 등이 들어 있습니다. 보통 파일 시스템에서 가장 하드 디스크 공간을 많이 차지하는 부분입니다. 최소한 500MB를 할당하십시오. 시스템에 설치할 패키지의 수와 종류에 따라 더 늘려야 할 수도 있습니다. 보통 워크스테이션이나 서버로 설치하려면 4–6GB 정도가 필요합니다.
- 이제 /usr 파티션은 루트 파티션(/)과 같이 두기를 권장합니다. 그러지 않으면 부팅할 때 문제가 생길 수도 있습니다. 그래서 루트 파티션은 /usr를 포함해 최소한 600–750MB의 디스크 공간을 확보해야 하고, 워크스테이션과 서버에서 5–6GB 정도를 확보해야 합니다.
- /var: 뉴스 기사, 전자 메일, 웹 페이지, 데이터베이스, 패키지 시스템의 캐시 등 자주 변하는 정보가 주로 저장됩니다. 이 디렉터리의 크기는 시스템의 이용 방법에 크게 좌우되지만 대부분의 시스템에서는 패키지 관리 도구의 사용분이 가장 큰 영향을 가지게 될 것입니다. 데비안이 제공하는 모든 것을 한꺼번에 전체 설치하는 경우에도 /var에 2 또는 3GB 정도를 할당하시면 충분합니다. 한 번에 모두 설치하지 않고 부분 부분을 서서히(예를 들면, 우선 서비스와 유틸리티, 다음에 콘솔용의 것, 다음에 X 용의것...과 같이) 설치하는 경우, 300–500MB의 여유 공간 있으면 좋습니다. 하드디스크의 빈 용량이 귀중하고, 대대적인 업데이트 예정이 없다면 30 또는 40MB 정도에서도 어떻게든 해 나갈 수 있습니다.

- /tmp: 프로그램이 만든 임시 데이터를 저장합니다. 40–100MB 정도면 충분합니다. 압축 유틸리티, CD/DVD 굽기 유틸리티, 멀티미디어 프로그램의 경우 이미지 파일을 /tmp에 임시로 저장하기도 합니다. 이러한 프로그램을 사용한다면 /tmp의 크기를 적절히 조절하십시오.

- /home: 모든 사용자는 이 디렉터리의 서브디렉터리에 개인 데이터를 저장합니다. 이 디렉터리의 크기는 이 시스템을 사용하는 사용자가 몇 명이고 디렉터리에 어떤 파일을 넣을 지에 따라 달라집니다. 예정된 사용량에 따라 다르지만, 각 사용자에게 100MB씩 할당하고, 필요에 따라 이 값을 조정하십시오. 홈 디렉터리에 다수의 멀티미디어 파일(사진, MP3, 동영상)을 저장할 예정이면 더 많은 용량을 잡아 주십시오.

C.3 권장하는 파티션 구조

신규 사용자와, 개인 데비안 사용자, 가정용 시스템, 기타 혼자서 사용하는 시스템의 경우, / 파티션 1개(그리고 swap 추가)로 끝내는 것이 가장 쉽고 간단한 방식입니다. 파티션 종류는 ext4를 권장합니다.

여러명이 사용하는 시스템이거나 하드디스크의 용량이 큰 시스템에서는 /var, /tmp, /home 각각을 / 파티션과는 별도의 파티션에 두는 것이 좋습니다.

데비안 배포판에 포함되지 않은 프로그램을 많이 설치할 계획이라면 /usr/local 파티션이 필요할지도 모릅니다. 또한 메일 서버로 사용한다면, /var/mail를 다른 파티션으로 할 필요가 있을지도 모릅니다. 여럿의 사용자 계정이 있는 서버를 준비하는 경우, 독립적이고 용량이 큰 /home 파티션을 준비하는 편이 대체로 좋습니다. 이렇게 파티션의 구성은 컴퓨터 상황에 따라 경우에 따라 다양합니다.

매우 복잡한 시스템의 경우, 멀티디스크 HOWTO를 참고하십시오. ISP나 서버 관리자가 관심있어할 만한 심도 있는 내용을 다루고 있습니다.

스왑 파티션의 크기에 대해서는 여러가지 생각이 다릅니다. 한 가지 방법은 시스템 메모리만큼 스왑 공간을 잡는 것입니다. 또 대부분의 경우 512MB보다 작으면 안 됩니다. 물론 이런 규칙에도 예외는 있습니다.

예를 들어 좀 오래된 컴퓨터에 램이 512MB 있고 /dev/sda에 20GB짜리 SATA 드라이브가 있을 수도 있습니ek. 그러면 8GB 파티션 /dev/sda1에 다른 운영체제가 설치되어 있고, 512MB 스왑 파티션을 /dev/sda3에 사용하고, 11.4GB 파티션 /dev/sda2를 리눅스에 사용할 수도 있습니다.

시스템 설치가 끝났을 때 얼마나 공간을 차지할 지 알고 싶으면, D.2절 부분을 참고하십시오.

C.4 리눅스의 장치 이름

리눅스에서 디스크와 파티션을 부르는 이름이 다른 운영 체제와 다르기도 합니다. 파티션을 만들고 파티션할 때 이 리눅스 이름을 알고 있어야 합니다. 기본적으로는 다음 규칙을 따릅니다:

- 첫번째 발견한 하드디스크의 이름은 /dev/sda입니다.

- 두번째 발견한 하드디스크의 이름은 /dev/sdb이고, 그 이후는 마찬가지입니다.

- 첫번째 SCSI CD-ROM은 /dev/scd0이라고 하고, /dev/sr0이라고도 합니다.

드라이브의 파티션 이름은 디스크 이름 뒤에 숫자를 붙입니다. sda1와 sda2는 각각 첫번째 SCSI 디스크의 첫번째와 두번째 파티션을 말합니다.

실제 예를 들어보면 다음과 같습니다. SCSI 디스크가 2개 있어서, 하나는 SCSI 주소 2에 연결되어 있고 다른 하나는 4에 연결되어 있습니다. 첫번째(2번 주소에 연결된) 디스크가 sda이고, 두번째(4번 주소에

연결된) 디스크가 sdb입니다. sda에 파티션이 3개이면, 그 파티션의 이름은 sda1, sda2, sda3입니다. sdb 디스크와 그 파티션도 같은 방식입니다.

SCSI 호스트 버스 어댑터(컨트롤러)가 2개 있으면 어느 드라이브가 첫번째가 될지 알기 어려울 수도 있습니다. 이 경우엔 부팅할 때 메시지를 잘 보고, 드라이브의 모델과 용량으로 파악하는 게 최선의 방법입니다.

C.5 데비안의 파티션 프로그램

여러가지 종류의 파티션 도구가 내장된 다양한 형식의 하드디스크나 시스템에서 작동하도록 데비안 개발자가 준비해 놓았습니다. 아래에 시스템에서 사용할 수 있는 프로그램의 목록을 나타냅니다.

partman 데비안 권장 파티션 도구입니다. 이 만능 프로그램은 파티션 크기를 변경하거나 파일 시스템을 만들거나 마운트 지점을 지정하거나 할 수 있습니다.

fdisk 처음부터 있던 리눅스 파티션 프로그램. 전문가용.

FreeBSD용 파티션이 디스크에 있으면 주의해야 합니다. 설치용 커널은 이 형식의 파티션을 지원하지만, **fdisk** 프로그램에서 이 파티션을 화면에 표시하는 형식이 다릅니다. 리눅스+FreeBSD 하우투를 참고하십시오.

cfdisk 간단하고 널리 사용하는 전체 화면 파티션 프로그램.

cfdisk는 FreeBSD 파티션을 전혀 인식하지 못하기 때문에(다시 말하지만) 장치의 이름이 다를 수도 있으니 주의하십시오.

디스크 파티션하기 메뉴를 선택하면 위 프로그램 중 하나를 실행합니다. VT2에서 명령행을 이용해 다른 파티션 도구를 사용할 수도 있지만, 이 방법은 추천하지 않습니다.

부록 D

여러가지 내용

D.1 리눅스 장치

리눅스에서는 /dev 디렉터리 아래에 여러가지 특수 파일이 들어 있습니다. 이 파일을 장치 파일이라고 하고, 이 파일은 일반 파일과는 다르게 동작합니다. 장치 파일 중에 가장 많은 종류가 블럭 장치와 캐릭터 장치에 대한 장치 파일입니다. 이 파일은 실제(리눅스 커널에 들어 있는) 드라이버에 대한 인터페이스 역할을 합니다. (그리고 리눅스 커널에 들어 있는 드라이버는 하드웨어에 접근합니다.) 흔하지는 않지만 또 다른 종류의 장치 파일이 있는데, 파이프라고 합니다. 아래 표에 중요한 장치 파일 몇 개가 쓰여 있습니다.

sda	첫번째 하드디스크
sdb	두번째 하드디스크
sda1	첫번째 하드디스크의 첫번째 파티션
sdb7	두번째 하드디스크의 일곱번째 파티션

sr0	첫번째 CD-ROM
sr1	두번째 CD-ROM

ttyS0	시리얼 포트 0, MS-DOS에서는 COM1
ttyS1	시리얼 포트 1, MS-DOS에서는 COM2
psaux	PS/2 마우스 장치
gpmdata	가짜 장치, GPM (마우스) 데몬에서 나온 데이터의 리피터

cdrom	CD-ROM 드라이브에 대한 심볼릭 링크
mouse	마우스 장치 파일에 대한 심볼릭 링크

null	이 장치로 들어가는 데이터는 모두 사라집니다
zero	이 장치에서 끊임없이 0을 읽을 수 있습니다

D.1.1 마우스 설정하기

마우스는 리눅스 콘솔과(gpm 사용) X 윈도우 환경 모두에서 사용할 수 있습니다. 보통 `gpm`과 X 서버 자체를 설치하기만 하면 마우스를 사용할 수 있습니다. 두 환경 모두 마우스 장치로 `/dev/input/mice`를 사용합니다. 마우스 프로토콜은 gpm에서는 **exps2**, X 환경에서는 **ExplorerPS/2**입니다. 설정 파일은 `/etc/gpm.conf`와 `/etc/X11/xorg.conf`입니다.

마우스를 사용하려면 특정 커널 모듈을 읽어들여야 할 수 있습니다. 대부분 올바른 모듈을 자동으로 찾아내지만, 예전 시리얼 마우스나 버스 마우스[1], 매우 오래된 컴퓨터의 마우스는 찾아내지 못할 수 있습니다. 여러가지 마우스 종류의 리눅스 커널 모듈은 아래 표에 있습니다:

모듈	설명
psmouse	PS/2 마우스(자동으로 찾아냄)
usbhid	USB 마우스(자동으로 찾아냄)
sermouse	대부분의 시리얼 마우스
logibm	Logitech 어댑터카드에 연결된 버스 마우스
inport	ATI나 마이크로소프트 InPort카드에 연결된 버스마우스

마우스 드라이버 모듈을 읽어들이려면 **modconf** 명령을(같은 이름의 패키지에 들어 있습니다) 사용할 수 있습니다. 모듈은 **kernel/drivers/input/mouse** 분류에 있습니다.

D.2 태스크마다 필요한 디스크 공간

모든 표준 패키지가 들어 있고 기본 커널을 사용하는 amd64 아키텍처의 표준 설치 용량은 971MB의 디스크 공간을 차지합니다. “표준 시스템” 태스크를 선택하지 않으면 최소의 베이스 시스템 설치는 769MB를 차지합니다.

중요

두 경우 모두, 설치가 끝나고 임시 파일을 지운 후에 실제 차지하는 디스크 용량입니다. 저널링 파일과 같이 파일 시스템에서 사용하는 오버헤드는 감안하지 않았습니다. 즉 이보다 더 큰 디스크 공간이 설치하는 도중에도 필요하고 시스템을 실제 사용할 때도 필요합니다.

다음 표는 aptitude에서 표시하는 값으로(tasksel에 들어 있는) 태스크에 필요한 용량입니다. 태스크 중에는 겹치는 부분이 있기 때문에 두 개의 태스크를 같이 설치하면 숫자를 합친 전체 크기보다는 작을 수도 있습니다.

기본값으로 설치 프로그램은 그놈 데스크톱 환경을 설치합니다. 하지만 특별한 설치 이미지를 사용하거나, 부팅한 다음에 원하는 데스크톱 환경을 지정하면 다른 데스크톱 환경을 선택할 수도 있습니다. (6.3.6.2절 참고.)

[1] 시리얼 마우스는 일반적으로 9핀 D형 커넥터를 사용하고 버스마우스는 8핀 둥근 커넥터를 사용합니다. PS/2마우스의 6핀 커넥터나 ADB 마우스의 4핀 커넥터와 혼동할 수 있습니다.

파티션의 크기를 결정할 때, 표준 설치의 크기에 다음 표에 있는 크기를 더해야 합니다. "설치 크기"에 들어있는 크기의 대부분은 /usr 및 /lib에서 차지합니다. "다운로드 크기"는(일시적으로) /var에 필요합니다.

태스크	**설치 크기(MB)**	**다운로드 크기(MB)**	**설치하는데 필요한 공간(MB)**
데스크톱 환경			
• 그놈(기본값)	2790	786	3576
• KDE 플라스마	4122	1212	5334
• Xfce	2187	621	2808
• LXDE	2271	653	2924
• MATE	2574	711	3285
• 시나몬	4197	1251	5448
웹 서버	44	11	55
SSH 서버	2	0	2

영어가 아닌 언어로 설치한다면 **tasksel**에서 자동으로 지역화 태스크를(해당 언어에 대한 태스크가 있다면) 설치합니다. 언어마다 필요한 공간이 다릅니다. 다운로드하고 설치하는데 최대 350MB까지의 공간이 있어야 합니다.

D.3 유닉스/리눅스 시스템에서 데비안 GNU/리눅스 설치하기

이 부분은 설명서의 다른 부분에 설명되어있는 ncurses 기반 메뉴 방식 설치 프로그램을 사용하지 않고 기존의 Unix · Linux 시스템에서 데비안 GNU/리눅스를 설치하는 방법을 설명합니다. 이 "크로스 설치" HOWTO는 Red Hat, Mandriva, SUSE에서 데비안 GNU/리눅스로 이동하는 사용자의 요구로 작성되었습니다. 여기서는 *nix 명령의 입력에 대해 숙지하고 파일 시스템을 조작할 수 있는 것이 전제가 되고 있습니다. 여기서는 #가 데비안 chroot에 입력된 명령을 보여주고 $는 사용자의 기존 시스템에서 입력되는 명령을 나타냅니다.

일단 새로운 데비안 시스템에 맞게 설정하기만 하면, 기존 사용자 데이터를(있다면) 옮겨 와서 계속 사용할 수 있습니다. 따라서 이것은 "다운 타임 없음"에서 데비안 GNU/리눅스 설치됩니다. 또한 이것은 여러가지 부팅 설치 미디어와 잘되지 않는 하드웨어에서 좋은 설치 방법입니다.

참고

대부분 수동으로 해야 하므로, 시스템의 대부분의 기본 설정을 직접 해야 할 수도 있습니다. 일반적인 설치를 할 경우보다 데비안 및 Linux에 대한 지식이 많이 필요합니다. 또 이렇게 설치해서 일반적인 설치와 똑같은 시스템이 될 것으로 기대할 수 없습니다. 또 시스템의 기본적인 단계에 지나지 않습니다. 추가로 설치 및 설정이 필요하게 될지도 모릅니다.

D.3.1 시작하기

기존 유닉스용 파티션 도구를 이용해 하드 드라이브를 필요한 대로 다시 파티션하십시오. 최소한 파일 시스템 한 개를 스왑으로 만드십시오. 콘솔만 설치하는 경우는 약 769MB의 공간이 필요하고 X를 설치한다면 약

2271MB가(그놈이나 KDE 플라즈마같은 데스크톱 환경을 설치한다면 이보다 더) 필요합니다.

그리고 파티션에 파일 시스템을 만드십시오. 예를 들어 /dev/sda6 파티션에 ext3 파일 시스템을 만드는 경우라면(여기 예제에서 루트 파티션입니다):

```
# mke2fs -j /dev/sda6
```

ext2 파일 시스템을 만드는 경우라면 **-j** 옵션을 빼십시오.

스왑을 다음과 같이 초기화하고 활성화하십시오(파티션 번호는 데비안 스왑 파티션에 파티션 번호로 바꾸십시오):

```
# mkswap /dev/sda5
# sync
# swapon /dev/sda5
```

파티션 /mnt/debinst (설치 지점. 새로운 시스템의 root (/) 파일 시스템에 있습니다)에 마운트하고 하십시오. 엄밀히 말하면 마운트 위치 이름은 아무거나 상관 없습니다. 이후의 설명에서 이것을 사용합니다.

```
# mkdir /mnt/debinst
# mount /dev/sda6 /mnt/debinst
```

참고

파일 시스템의 일부를(예를 들어 /usr) 별도의 파티션에 마운트하려면, 다음 단계로 넘어가기 전에 그 디렉터리를 수동으로 만들어서 마운트해야 합니다.

D.3.2 debootstrap 설치

데비안 설치 프로그램에서 사용하는 유틸리티에서 데비안베이스 시스템을 설치하는 공식적인 방법으로 인정받고 있는 것은 **debootstrap**입니다. **wget**와 **ar**를 사용하지만 /bin/sh와 기본적인 Unix/Linux 도구[2]에만 의존하고 있습니다. 기존 시스템에 아직 설치되어 있지 않으면 **wget**와 **ar**를 설치한 다음 **debootstrap** 다운로드 설치하십시오.

아니면, 수동으로 설치하려면 다음과 같이 합니다. deb 파일을 풀 작업 폴더를 다음과 같이 만드십시오:

```
# mkdir work
# cd work
```

debootstrap 바이너리는 데비안 아카이브(아키텍쳐에 맞는 파일을 선택)에 있습니다. pool에서 **debootstrap** deb 파일을 다운로드하고, 작업 폴더에 패키지를 복사하고, 파일을 추출합니다. 파일을 설치하려면 root 권한이 필요할 수도 있습니다.

```
# ar -x debootstrap_0.X.X_all.deb
# cd /
# zcat /full-path-to-work/work/data.tar.gz | tar xv
```

[2]에는 **sed**, **grep**, **tar**, **gzip** 같은 GNU 핵심 유틸리티가 들어 있습니다.

D.3.3 debootstrap 실행

debootstrap를 실행하면 아카이브에서 필요한 파일을 직접 다운로드할 수 있습니다. 다음 명령 예제에서는 **ftp.kr.debian.org/debian**하고 있지만 네트워크에서 가까운 데비안 아카이브 미러 사이트를 입력할 수 있습니다. 미러 사이트는 http://www.debian.org/mirror/list에 목록이 있습니다.

bullseye 데비안 GNU/리눅스 설치 이미지를 /cdrom에 마운트했다면 http URL 대신에 file URL을 쓸 수 있습니다: **file:/cdrom/debian/**

debootstrap 명령에서 ARCH를 다음 중의 하나로 바꾸십시오: **amd64**, **arm64**, **armel**, **armhf**, **i386**, **mips64el**, **mipsel**, **ppc64el**, **s390x**.

```
# /usr/sbin/debootstrap --arch ARCH bullseye \
     /mnt/debinst http://ftp.us.debian.org/debian
```

대상 아키텍쳐가 호스트와 다르면, **--foreign** 옵션을 붙여야 합니다.

D.3.4 베이스 시스템 설정

이제 디스크에 진정한 데비안 시스템을(많이 작지만) 손에 넣었습니다. 거기에 **chroot**하십시오:

```
# LANG=C.UTF-8 chroot /mnt/debinst /bin/bash
```

타겟 아키텍쳐가 호스트와 다르면, 먼저 qemu-user-static을 새 호스트에 복사해야 합니다:

```
# cp /usr/bin/qemu-ARCH-static /mnt/debinst/usr/bin
# LANG=C.UTF-8 chroot /mnt/debinst qemu-ARCH-static /bin/bash
```

chroot 후, 데비안 기본 시스템과 호환되는 터미널 정의가 필요할 수 있습니다. 예를 들어 다음과 같이합니다.

```
# export TERM=xterm-color
```

TERM 값에 따라 ncurses-term 패키지를 설치해야 할 수도 있습니다.

타겟 아키텍쳐가 호스트와 다르면, 멀티스테이지 부팅 단계를 마쳐야 합니다:

```
/debootstrap/debootstrap --second-stage
```

D.3.4.1 장치 파일 만들기

이렇게 하면 /dev/에는 아주 기초적인 장치 파일만 들어 있게 됩니다. 다음 단계로 진행하려면 장치 파일이 몇 개 더 필요합니다. 여러가지 방법이 있고, 이 중에 어떤 방법을 이용할 지는 설치에 사용하는 호스트 시스템이 무엇이냐에 따라, 그리고 모듈식 커널을 이용할 것인가 아닌가, 그리고 새 시스템에 동적인(예를 들어 udev 사용) 장치 파일을 사용할 지 고정 장치 파일을 사용할 지에 따라 달라집니다.

사용할 수 있는 옵션 몇 가지를 설명하면:

- makedev 패키지를 설치하고, 다음 명령으로 기본적인 고정 장치 파일의 기본 모음을(chroot 상태에서) 만듭니다

  ```
  # apt install makedev
  # mount none /proc -t proc
  # cd /dev
  # MAKEDEV generic
  ```

- **MAKEDEV**를 이용해 수동으로 장치 파일을 직접 지정해서 만듭니다
- 호스트 시스템의 /dev를 대상 시스템의 /dev 디렉터리에 연결합니다. 어떤 패키지는 postinst 스크립트를 실행하면서 장치 파일을 만들 수도 있습니다. 그러므로 이 옵션은 주의해서 사용해야 합니다.

D.3.4.2 파티션 마운트하기

/etc/fstab를 만들어야 합니다.

```
# editor /etc/fstab
```

다음 예제를 필요에 맞게 편집할 수 있습니다.

```
 # /etc/fstab: static file system information.
#
# file system    mount point   type    options                 dump pass
/dev/XXX         /             ext3    defaults                0    1
/dev/XXX         /boot         ext3    ro,nosuid,nodev         0    2

/dev/XXX         none          swap    sw                      0    0
proc             /proc         proc    defaults                0    0

/dev/cdrom       /media/cdrom  iso9660 noauto,ro,user,exec     0    0

/dev/XXX         /tmp          ext3    rw,nosuid,nodev         0    2
/dev/XXX         /var          ext3    rw,nosuid,nodev         0    2
/dev/XXX         /usr          ext3    rw,nodev                0    2
/dev/XXX         /home         ext3    rw,nosuid,nodev         0    2
```

/etc/fstab에서 지정한 파일 시스템을 모두 마운트 **mount -a**라고 합니다. 또한 파일 시스템을 하나하나 마운트하려면 다음과 같이하십시오:

```
# mount /path   # e.g.: mount /usr
```

현재 데비안 시스템에서 이동식 미디어의 마운트 지점을 /media하고 있지만, /에 심볼릭 링크를 호환 유지하고 있습니다. 다음 예제와 같이 필요한 경우 작성하십시오:

```
 # cd /media
# mkdir cdrom0
# ln -s cdrom0 cdrom
# cd /
# ln -s media/cdrom
```

proc 파일 시스템은 어디서나 몇 번이라도 장착할 수 있지만, 관습으로 /proc에 마운트합니다. **mount -a**를 사용하지 않으면 다음과 같이 진행하기 전에 꼭 proc을 마운트하십시오.

```
# mount -t proc proc /proc
```

ls /proc 명령을 실행하면 여러 파일이 들어 있는 디렉터리 내용을 표시합니다. 이 명령이 실패하면 chroot 바깥에서 proc을 마운트할 수 있습니다:

```
# mount -t proc proc /mnt/debinst/proc
```

D.3.4.3 시간대 설정하기

/etc/adjtime 파일의 3번째 줄을 “UTC”로 설정하면 하드웨어 시계 값을 UTC로 해석하고, “LOCAL”로 설정하면 로컬 시각으로 해석합니다. 다음 명령어로 설정할 수 있습니다.

```
# editor /etc/adjtime
```

예를 들어 다음과 같이 합니다:

```
0.0 0 0.0
0
UTC
```

다음 명령으로 표준 시간대를 설정합니다.

```
# dpkg-reconfigure tzdata
```

D.3.4.4 네트워크 설정하기

”32-bit hard-float ARMv7에서는 현재 실험 버전 네트워크 설정을하려면, /etc/network/interfaces, /etc/resolv.conf, /etc/hostname과 /etc/hosts을 편집하십시오.

```
# editor /etc/network/interfaces
```

다음은 /usr/share/doc/ifupdown/examples 간단한 예입니다:

```
 ###################################################################
# /etc/network/interfaces -- configuration file for ifup(8), ifdown(8)
# See the interfaces(5) manpage for information on what options are
# available.
###################################################################

# 루프백 인터페이스는 이제는 실제 필요가 없지만, 필요한 경우 사용할
# 수 있습니다.
#
# auto lo
# iface lo inet loopback

# DHCP를 사용하려면:
#
# auto eth0
# iface eth0 inet dhcp

# 고정 IP 설정 예제: (network, broadcast, gateway는 안 써도 됩니다)
#
# auto eth0
# iface eth0 inet static
#     address 192.168.0.42
#     network 192.168.0.0
#     netmask 255.255.255.0
```

```
#     broadcast 192.168.0.255
#     gateway 192.168.0.1
```

/etc/resolv.conf에 이름 서버와 search 명령을 입력하십시오:

```
# editor /etc/resolv.conf
```

다음은 /etc/resolv.conf의 간단한 예입니다:

```
search hqdom.local
nameserver 10.1.1.36
nameserver 192.168.9.100
```

시스템의 호스트 이름(2글자에서 63 글자까지)를 입력하십시오:

```
 # echo DebianHostName > /etc/hostname
```

또한 IPv6를 지원하는 기본적인 /etc/hosts는 다음과 같이합니다:

```
 127.0.0.1 localhost
127.0.1.1 DebianHostName

# IPv6가 가능한 호스트에서는 다음 줄을 쓰는 게 좋습니다
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
```

여러 네트워크 카드를 가지고있다면 /etc/modules 파일에 원하는 순서로 드라이버 모듈 이름을 배치하십시오. 그래야 부팅할 때 각 카드가 의도한 해당 인터페이스 베이스 이름으로(eth0, eth1 등) 연결됩니다.

D.3.4.5 APT 설정하기

debootstrap은 아주 기본적인 /etc/apt/sources.list 파일을 만드므로 추가 패키지를 설치할 수 있습니다. 하지만 이 외에 소스를 추가해야 할 경우가 있습니다. 예를 들어 보안 업데이트의 소스 패키지를 설정할 수 있습니다:

```
deb-src http://ftp.us.debian.org/debian bullseye main

deb http://security.debian.org/ bullseye-security main
deb-src http://security.debian.org/ bullseye-security main
```

sources.list 파일을 고친 다음에 꼭 **apt update**를 실행하십시오.

D.3.4.6 로캘 및 키보드 설정하기

영어가 아닌 언어를 사용할 때 로캘을 설정하려면 locales 지원 패키지를 설치하고 그 패키지를 설정하십시오. 지금은 UTF-8 로캘 사용을 권장합니다:

```
# apt install locales
# dpkg-reconfigure locales
```

키보드를 설정하려면(키보드 설정이 필요한 경우):

```
# apt install console-setup
# dpkg-reconfigure keyboard-configuration
```

chroot 안에서는 키보드를 설정할 수 없으니 유의하십시오. 다시 시작한 다음에 설정합니다.

D.3.5 커널 설치

이 시스템을 시작할 수 있도록 한다면, Linux 커널과 부트 로더가 필요합니다. 다음과 같이하여 패키지로 만든 커널을 확인하십시오:

```
# apt search linux-image
```

다음 패키지 이름을 사용하려면 커널 패키지를 설치합니다.

```
# apt install linux-image-arch-etc
```

D.3.6 부트로더 설정하기

데비안 GNU/리눅스 시스템을 부팅 가능하게 만드려면, 설치한 커널과 루트 파티션을 읽어들이도록 부트로더를 설치하십시오. **debootstrap**은 부트 로더를 설치하지 않으니 주의하십시오. 설치하는데 데비안 chroot 내부 **apt**를 사용할 수 있습니다.

앞서 /dev/sda 장치 파일을 만들었다고 가정합니다. **grub2**를 설치하는 다른 방법도 있지만, 이 부록이 다룰 범위를 벗어나는 내용입니다.

D.3.7 원격 접근: SSH 설치 및 접근 설정

콘솔을 통해 시스템에 로그인할 수 있으면, 이 부분을 넘어가도 됩니다. 네트워크를 통해 시스템에 접근해야 하는 경우, SSH를 설치하고 접근을 설정해야 합니다.

```
# apt install ssh
```

암호를 사용한 루트 로그인은 기본적으로 막혀 있습니다. 그러니 접근 설정은 암호를 설정하고 암호를 사용한 루트 로그인을 열어 주면 됩니다:

```
# passwd
# editor /etc/ssh/sshd_config
```

다음 옵션을 사용해야 합니다:

```
PermitRootLogin yes
```

루트 계정에 ssh 키를 설정해도 됩니다:

```
# mkdir /root/.ssh
# cat << EOF > /root/.ssh/authorized_keys
ssh-rsa ....
EOF
```

마지막으로 루트가 아닌 사용자를 추가하고 암호를 설정해서 접근을 설정할 수 있습니다.

```
# adduser joe
# passwd joe
```

D.3.8 마지막 처리

앞에서 말한 것처럼, 설치한 시스템은 아주 기초적인 시스템입니다. 시스템을 좀 더 괜찮게 만드려면, 쉬운 방법으로 "standard" 우선 순위의 모든 패키지를 설치하면 됩니다:

```
# tasksel install standard
```

물론 **apt**를 이용해 패키지를 하나하나 선택해서 설치할 수도 있습니다.

설치한 다음에 /var/cache/apt/archives/ 밑에 다운로드한 패키지가 많이 들어 있게 됩니다. 다음 명령을 실행하면 디스크 공간을 좀 더 확보할 수 있습니다:

```
apt clean
```

D.4 PPP 오버 이더넷을(PPPoE) 이용해 데비안 GNU/리눅스 설치하기

어떤 국가에서는 PPP 오버 이더넷(PPPoE)이 초고속 인터넷 연결에서(ADSL 혹은 케이블) 인터넷 서비스 제공자에게 연결하는 일반적인 프로토콜입니다. PPPoE 연결은 기본값으로는 지원하지 않지만 아주 간단히 동작하게 만들 수 있습니다. 여기서 그 방법을 설명합니다.

설치할 때 설정한 PPPoE 연결은 설치한 시스템을 다시 시작한 다음에도 사용할 수 있습니다. (7장 참고.)

설치할 때 PPPoE를 설정하고 사용하는 옵션을 사용하려면, CD-ROM/DVD 이미지중 하나를 사용해야 합니다. 다른 설치 방법에서는 지원하지 않습니다. (예를 들어 netboot에서는 지원하지 않습니다.)

PPPoE를 통한 설치는 다른 설치와 거의 동일합니다. 아래에서 다른 부분을 설명합니다.

- 부팅 파라미터로 **modules=ppp-udeb**을 사용해 설치 프로그램을 부팅하십시오. 이렇게 하면 자동으로 PPPoE 설정을 하는 컴포넌트를 (ppp-udeb) 읽어들여서 실행합니다.
- 마찬가지로 설치 처음 단계를 계속 하십시오. (언어, 국가 및 키보드 선택. 그리고 필요한 경우 설치 프로그램 컴포넌트를 추가로 읽어들이기[3].)
- 다음 단계는 네트워크 하드웨어 찾기입니다. 시스템에 들어 있는 모든 이더넷 카드를 찾습니다.
- 그 다음에 실제로 PPPoE 설정을 시작합니다. 설치 프로그램에서 검색한 모든 이더넷 장치에 대해서 PPPoE 콘센트레이터(PPPoE 연결을 처리하는 서버)가 있는 지 찾아 봅니다.

[3] ppp-udeb 컴포넌트를 이 단계에서 추가 컴포넌트로 읽어들입니다. 중간이나 낮은 우선 순위로 설치한다면(전문가 모드), 부팅 프롬프트에서 "modules" 파라미터를 설정하지 않고 ppp-udeb을 선택할 수 있습니다.

첫번째 시도할 때 콘센트레이터를 찾지 못하는 경우도 있습니다. 네트워크가 느리거나 너무 로드가 심하거나 서버에 문제가 있는 경우 이런 일이 발생할 수 있습니다. 대부분의 경우 다시 한번 콘센트레이터를 검색해 보면 성공합니다. 다시 시도해 보려면 설치 프로그램의 메인 메뉴에서 PPPoE 연결 설정 및 시작을 선택하십시오.

- 콘센트레이터를 찾으면, 로그인 정보를(PPPoE 사용자 이름 및 암호) 입력할 수 있게 물어봅니다.
- 여기서 설치 프로그램은 입력한 정보를 이용해 PPPoE에 연결합니다. 올바른 정보를 입력했다면, PPPoE 연결을 설정하고 PPPoE를 이용해 인터넷에 연결해(필요한 경우) 패키지를 인터넷에서 받아올 수 있게 됩니다. 로그인 정보가 틀렸거나 기타 오류가 발생한 경우에는 설치 프로그램이 멈춥니다. 하지만 PPPoE 연결 설정 및 시작을 선택하면 다시 설정을 할 수 있습니다.

부록 E

문서 관리 정보

E.1 문서 정보

이 설명서는 초기 데비안 설치 설명서를 바탕으로 한 boot-floppies의 Woody 설치 설명서를 바탕으로 사지(Sarge)의 debian-installer를 위해 작성되었습니다. 또한 2003년에 GPL로 발표된 Progeny 배포판 설명서에 기반하고 있습니다.

이 문서는 닥북(DocBook) XML 형식으로 작성되어 있습니다. `docbook-xml`과 `docbook-xsl` 패키지에 있는 정보를 이용해서 여러가지 프로그램이 문서의 여러가지 형식의 출력을 만들어 냅니다.

문서를 유지보수하기 좋도록 엔티티와 프로파일 속성과 같은 여러가지 XML 기능을 이용합니다. 엔티티와 속성은 프로그래밍 언어의 변수 및 조건문과 비슷한 역할을 합니다. 이 문서의 XML 소스에는 여러가지 아키텍처에 대한 정보가 모두 들어 있고, 각 아키텍처에 해당하는 텍스트를 분리하는 데 프로파일 속성을 사용합니다.

이 문서의 한국어 번역에 참여한 사람은 다음과 같습니다. (가나다순) 류창우, 박선재, 이경순, 이광우, 이유미, 장석문, 최병현. 도움을 주신 모든 분에게 감사드립니다.

E.2 이 문서에 참여하기

이 문서에 대해 문제점이나 의견이 있으면 `installation-guide` 패키지를 이용해 버그리포트를 보내주십시오. `reportbug` 패키지를 참고하시고, 데비안 버그 추적 시스템 온라인 문서를 읽어보십시오. 해당 문제점이 벌써 보고된 상태인지 알아보려면 installation-guide에 해결 안 된 버그 목록을 보는 것도 좋습니다. 이미 보고된 버그인 경우, `XXXX`@bugs.debian.org 주소에 추가로 보강할 만한 정보나 도움이 될 만한 정보를 메일로 보낼 수 있습니다. 여기서 `XXXX`는 보고한 버그의 번호입니다.

더 좋은 방법으로, 이 문서의 닥북 소스 코드를 구해서 패치를 만드십시오. 닥북 소스 코드는 salsa의 installation-guide 프로젝트에 있습니다. 닥북에 익숙하지 않더라도 걱정하지 마십시오. 설명서 디렉터리에 간단한 쪽지가 있고 이 쪽지를 읽는 걸로 시작하십시오. 닥북은 HTML과 비슷하면서도 텍스트의 화면 표시보다 의미에 중점을 두는 형식입니다. 패치는(아래에 있는) debian-boot 메일링 리스트로 보내주십시오. SVN으로 소스 코드를 받는 방법은, 소스 코드가 있는 맨 위 디렉터리에서 README 파일을 보십시오.

절대로 이 문서의 저자에게 직접 연락하지 마십시오. `debian-installer`에 대한 토론을 하는(이 설명서에 대한 토론 포함) 리스트가 있습니다. 이 메일링 리스트는 debian-boot@lists.debian.org입니다. 이 리스트에 가입하는 방법은 데비안 메일링 리스트 가입 페이지에 있고, 데비안 메일링 리스트 아카이브를 온라인으로 볼 수 있습니다.

E.3 중요 기여자들

이 문서는 브루스 페렌스(Bruce Perens), 스벤 루돌프(Sven Rudolph), 이고르 그로브먼(Igor Grobman), 제임스 트리시(James Treacy), 그리고 아담 디 카를로(Adam Di Carlo)가 작성했습니다. 세바스찬 레이(Sebastian Ley)는 설치 하우투를 썼습니다.

미로슬라브 쿠르제(Miroslav Ku⌧e)씨는 사지(Sarge)의 debian-installer에 많은 새로운 기능을 문서화했습니다.

매우 많은 데비안 사용자와 개발자가 이 문서에 기여하고 있습니다. 특히 다양한 문서를 편집 저술하고 있는 마이클 슈미츠(Michael Schmitz) (m68k 지원), 프랭크 노이만(Frank Neumann) (Amiga install manual의 원저자), 아르토 아스탈라(Arto Astala), 에릭 들루니(Eric Delaunay) / 벤 콜린스(Ben Collins) (SPARC 정보), 타피오 레토넨(Tapio Lehtonen), 스테판 보르츠메이어(Stéphane Bortzmeyer)에는 상당한 협력을 받았습니다. 또한 USB 메모리로 부팅하는 방법에 대한 유용한 정보를 주신 파스칼 르 베일(Pascal Le Bail)에 감사드립니다.

매우 큰 도움이 되었던 글과 정보는 다음 문서에 들어 있습니다: 짐 민타(Jim Mintha)의 네트워크 부팅에 관한 하우투(URL 없음), Debian FAQ, Linux/m68k FAQ, Linux for SPARC Processors FAQ, Linux/Alpha FAQ. 자유롭게 사용할 수 있고 풍부한 정보가 들어 있는 이 문서의 관리자들에게 깊은 감사를 표합니다.

이 설명서에서 chroot을 이용한 설치에 관한 부분은 (D.3절) 카르스텐 M. 셀프(Karsten M. Self)에 저작권이 있는 문서의 일부분에서 가져왔습니다.

E.4 상표권 안내

모든 상표는 그 상표권자의 소유입니다.

부록 F

GNU 일반 공중 사용 허가서

참고

This is an unofficial translation of the GNU General Public License into Korean language. It was not published by the Free Software Foundation, and does not legally state the distribution terms for software that uses the GNU GPL — only the original English text of the GNU GPL does that. However, we hope that this translation will help Korean speakers to better understand the GNU GPL.

이 문서는 GNU General Public License의 한국어 번역입니다. 이 번역문은 자유 소프트웨어 재단이 발행한 문서가 아니고, GNU GPL 소프트웨어의 배포조건에 대해 법적인 효력이 없습니다. GNU GPL의 영어로 된 원문 텍스트만이 효력을 가집니다. 이 번역문은 한국어 사용자가 GNU GPL을 더 쉽게 이해하기 위한 용도입니다.

2판, 1991년 6월

F.1 전문

소프트웨어에 적용되는 대부분의 사용 허가서(license)들은 소프트웨어에 대한 수정과 공유의 자유를 제한하려는 것을 그 목적으로 합니다. 그러나 GNU 일반 공중 사용 허가서(이하, "GPL"이라고 칭합니다.)는 자유 소프트웨어에 대한 수정과 공유의 자유를 모든 사용자들에게 보장하기 위해서 성립된 것입니다. 자유 소프트웨어 재단이 제공하는 대부분의 소프트웨어들은 GPL에 의해서 관리되고 있으며, 몇몇 소프트웨어에는 별도의 사용 허가서인 GNU 라이브러리 일반 공중 사용 허가서(GNU Library General Public License)를 대신 적용하기도 합니다. 자유 소프트웨어란, 이를 사용하려고 하는 모든 사람에 대해서 동일한 자유와

권리가 함께 양도되는 소프트웨어를 말하며 프로그램 저작자의 의지에 따라 어떠한 종류의 프로그램에도 GPL을 적용할 수 있습니다. 따라서 여러분이 만든 프로그램에도 GPL을 적용할 수 있습니다.

자유 소프트웨어를 언급할 때 사용되는 "자유"라는 단어는 무료(無料)를 의미하는 금전적인 측면의 자유가 아니라 구속되지 않는다는 관점에서의 자유를 의미하며, GPL은 자유 소프트웨어를 이용한 복제와 개작, 배포와 수익 사업 등의 가능한 모든 형태의 자유를 실질적으로 보장하고 있습니다. 여기에는 원시 코드(source code)의 전부 또는 일부를 원용해서 개선된 프로그램을 만들거나 새로운 프로그램을 창작할 수 있는 자유가 포함되며, 자신에게 양도된 이러한 자유와 권리를 보다 명확하게 인식할 수 있도록 하기 위한 규정도 포함되어 있습니다.

GPL은 GPL 안에 소프트웨어를 양도받을 사용자의 권리를 제한하는 조항과 단서를 별항으로 추가시키지 못하게 함으로써 사용자들의 자유와 권리를 실제적으로 보장하고 있습니다. 자유 소프트웨어의 개작과 배포에 관계하고 있는 사람들은 이러한 무조건적인 권리 양도 규정을 준수해야만 합니다.

예를 들어 GPL 프로그램을 배포할 경우에는 프로그램의 유료 판매나 무료 배포에 관계없이 자신이 해당 프로그램에 대해서 가질 수 있었던 모든 권리를, 프로그램을 받게될 사람에게 그대로 양도해 주어야 합니다. 이 경우, 프로그램의 원시 코드를 함께 제공하거나 원시 코드를 구할 수 있는 방법을 확실히 알려주어야 하고 이러한 모든 사항들을 사용자들이 분명히 알 수 있도록 명시해야 합니다.

자유 소프트웨어 재단은 다음과 같은 두 가지 단계를 통해서 사용자들을 권리를 보호합니다. (1) 소프트웨어에 저작권을 설정합니다. (2) 저작권의 양도에 관한 실정법에 의해서 유효한 법률적 효력을 갖는 GPL을 통해 소프트웨어를 복제하거나 개작 및 배포할 수 있는 권리를 사용자들에게 부여합니다.

자유 소프트웨어를 사용하는 사람들은 반복적인 재배포 과정을 통해 소프트웨어 자체에 수정과 변형이 일어날 수도 있으며, 이는 최초의 저작자가 만든 소프트웨어가 갖고 있는 문제가 아닐 수 있다는 개연성을 인식하고 있어야 합니다. 우리는 개작과 재배포 과정에서 다른 사람에 의해 발생된 문제로 인해 프로그램 원저작자들의 신망이 훼손되는 것을 원하지 않습니다. GPL에 자유 소프트웨어에 대한 어떠한 형태의 보증도 규정하지 않는 이유는 이러한 점들이 고려되었기 때문이며, 이는 프로그램 원저작자와 자유 소프트웨어 재단의 자유로운 활동을 보장하는 현실적인 수단이기도 합니다.

특허 제도는 자유 소프트웨어의 발전을 위협하는 요소일 수밖에 없습니다. 자유 프로그램을 재배포하는 사람들이 개별적으로 특허를 취득하게 되면, 결과적으로 그 프로그램이 독점 소프트웨어가 될 가능성이 있습니다. 자유 소프트웨어 재단은 이러한 문제에 대처하기 위해서 어떠한 특허에 대해서도 그 사용 권리를 모든 사람들(이하, "공중(公衆)"이라고 칭합니다.)에게 자유롭게 허용하는 경우에 한해서만 자유 소프트웨어와 함께 사용할 수 있다는 것을 명확히 밝히고 있습니다.

복제(copying)와 개작(modification) 및 배포(distribution)에 관련된 구체적인 조건과 규정은 다음과 같습니다.

F.2 GNU 일반 공중 사용 허가서 (GNU GENERAL PUBLIC LICENSE)

복제와 개작 및 배포에 관한 조건과 규정

제 0 조. 본 허가서는 GNU 일반 공중 사용 허가서의 규정에 따라 배포될 수 있다는 사항이 저작권자에 의해서 명시된 모든 컴퓨터 프로그램 저작물에 대해서 동일하게 적용됩니다. 컴퓨터 프로그램 저작물(이하, "프로그램"이라고 칭합니다.)이란 특정한 결과를 얻기 위해서 컴퓨터 등의 정보 처리 능력을 가진 장치(이하, "컴퓨터"라고 칭합니다.) 내에서 직접 또는 간접으로 사용되는 일련의 지시 및 명령으로 표현된 창작물을 의미하고, "2차적 프로그램"이란 전술한 프로그램 자신 또는 저작권법의 규정에 따라 프로그램의 전부 또는 상당 부분을 원용하거나 다른 언어로의 번역을 포함할 수 있는 개작 과정을 통해서 창작된 새로운 프로그램과 이와 관련된 저작물을 의미합니다. (이후로 다른 언어로의 번역은 별다른 제한없이 개작의 범위에 포함되는

것으로 간주합니다.) “피양도자”란 GPL의 규정에 따라 프로그램을 양도받은 사람을 의미하고, “원(原)프로그램”이란 프로그램을 개작하거나 2차적 프로그램을 만들기 위해서 사용된 최초의 프로그램을 의미합니다.

본 허가서는 프로그램에 대한 복제와 개작 그리고 배포 행위에 대해서만 적용됩니다. 따라서 프로그램을 실행시키는 행위에 대한 제한은 없습니다. 프로그램의 결과물(output)에는, 그것이 프로그램을 실행시켜서 생성된 것인지 아닌지의 여부에 상관없이 결과물의 내용이 원프로그램으로부터 파생된 2차적 프로그램을 구성했을 때에 한해서 본 허가서의 규정들이 적용됩니다. 2차적 프로그램의 구성 여부는 2차적 프로그램 안에서의 원프로그램의 역할을 토대로 판단합니다.

제 1 조. 적절한 저작권 표시와 프로그램에 대한 보증이 제공되지 않는다는 사실을 각각의 복제물에 명시하는 한, 피양도자는 프로그램의 원시 코드를 자신이 양도받은 상태 그대로 어떠한 매체를 통해서도 복제하고 배포할 수 있습니다. 복제와 배포가 이루어 질 때는 본 허가서와 프로그램에 대한 보증이 제공되지 않는다는 사실에 대해서 언급되었던 모든 내용을 그대로 유지시켜야 하며, 영문판 GPL을 함께 제공해야 합니다.

배포자는 복제물을 물리적으로 인도하는데 소요된 비용을 청구할 수 있으며, 선택 사항으로 독자적인 유료 보증을 설정할 수 있습니다.

제 2 조. 피양도자는 자신이 양도받은 프로그램의 전부나 일부를 개작할 수 있으며, 이를 통해서 2차적 프로그램을 창작할 수 있습니다. 개작된 프로그램이나 창작된 2차적 프로그램은 다음의 사항들을 모두 만족시키는 조건에 한해서, 제1조의 규정에 따라 또다시 복제되고 배포될 수 있습니다.

a. 파일을 개작할 때는 파일을 개작한 사실과 그 날짜를 파일 안에 명시해야 합니다.

b. 배포하거나 공표하려는 저작물의 전부 또는 일부가 양도받은 프로그램으로부터 파생된 것이라면, 저작물 전체에 대한 사용 권리를 본 허가서의 규정에 따라 공중에게 무상으로 허용해야 합니다.

c. 개작된 프로그램의 일반적인 실행 형태가 대화형 구조로 명령어를 읽어 들이는 방식을 취하고 있을 경우에는, 적절한 저작권 표시와 프로그램에 대한 보증이 제공되지 않는다는 사실, (별도의 보증을 설정한 경우라면 해당 내용) 그리고 양도받은 프로그램을 본 규정에 따라 재배포할 수 있다는 사실과 GPL 사본을 참고할 수 있는 방법이 함께 포함된 문구가 프로그램이 대화형 구조로 평이하게 실행된 직후에 화면 또는 지면으로 출력되도록 작성되어야 합니다. (예외 규정: 양도받은 프로그램이 대화형 구조를 갖추고 있다 하더라도 통상적인 실행 환경에서 전술한 사항들이 출력되지 않는 형태였을 경우에는 이를 개작한 프로그램 또한 관련 사항들을 출력시키지 않아도 무방합니다.)

위의 조항들은 개작된 프로그램 전체에 적용됩니다. 만약, 개작된 프로그램에 포함된 특정 부분이 원프로그램으로부터 파생된 것이 아닌 별도의 독립 저작물로 인정될 만한 상당한 이유가 있을 경우에는 해당 저작물의 개별적인 배포에는 본 허가서의 규정들이 적용되지 않습니다. 그러나 이러한 저작물이 2차적 프로그램의 일부로서 함께 배포된다면 개별적인 저작권과 배포 기준에 상관없이 저작물 모두에 본 허가서가 적용되어야 하며, 전체 저작물에 대한 사용 권리는 공중에게 무상으로 양도됩니다.

이러한 규정은 개별적인 저작물에 대한 저작자의 권리를 침해하거나 인정하지 않으려는 것이 아니라, 원프로그램으로부터 파생된 2차적 프로그램이나 수집 저작물의 배포를 일관적으로 규제할 수 있는 권리를 행사하기 위한 것입니다.

원프로그램이나 원프로그램으로부터 파생된 2차적 프로그램을 이들로부터 파생되지 않은 다른 저작물과 함께 단순히 저장하거나 배포할 목적으로 동일한 매체에 모아 놓은 집합물의 경우에는, 원프로그램으로부터 파생되지 않은 다른 저작물에는 본 허가서의 규정들이 적용되지 않습니다.

제 3 조. 피양도자는 다음 중 하나의 항목을 만족시키는 조건에 한해서 제1조와 제2조의 규정에 따라 프로그램(또는 제2조에서 언급된 2차적 프로그램)을 목적 코드(object code)나 실행물(executable form)의 형태로 복제하고 배포할 수 있습니다.

a. 목적 코드나 실행물에 상응하는 컴퓨터가 인식할 수 있는 완전한 원시 코드를 함께 제공해야 합니다. 원시 코드는 제1조와 제2조의 규정에 따라 배포될 수 있어야 하며, 소프트웨어의 교환을 위해서 일반적으로 사용되는 매체를 통해 제공되어야 합니다.

b. 배포에 필요한 최소한의 비용만을 받고 목적 코드나 실행물에 상응하는 완전한 원시 코드를 배포하겠다는, 최소한 3년간 유효한 약정서를 함께 제공해야 합니다. 이 약정서는 약정서를 갖고 있는 어떠한 사람에 대해서도 유효해야 합니다. 원시 코드는 컴퓨터가 인식할 수 있는 형태여야 하고 제1조와 제2조의 규정에 따라 배포될 수 있어야 하며, 소프트웨어의 교환을 위해서 일반적으로 사용되는 매체를 통해 제공되어야 합니다.

c. 목적 코드나 실행물에 상응하는 원시 코드를 배포하겠다는 약정에 대해서 자신이 양도받은 정보를 함께 제공해야 합니다. (제3항은 위의 제2항에 따라 원시 코드를 배포하겠다는 약정을 프로그램의 목적 코드나 실행물과 함께 제공 받았고, 동시에 비상업적인 배포를 하고자 할 경우에 한해서만 허용됩니다.)

저작물에 대한 원시 코드란 해당 저작물을 개작하기에 적절한 형식을 의미합니다. 실행물에 대한 완전한 원시 코드란 실행물에 포함된 모든 모듈들의 원시 코드와 이와 관련된 인터페이스 정의 파일 모두, 그리고 실행물의 컴파일과 설치를 제어하는데 사용된 스크립트 전부를 의미합니다. 그러나 특별한 예외의 하나로서, 실행물이 실행될 운영체제의 주요 부분(컴파일러나 커널 등)과 함께 (원시 코드나 바이너리의 형태로) 일반적으로 배포되는 구성 요소들은 이러한 구성 요소 자체가 실행물에 수반되지 않는 한 원시 코드의 배포 대상에서 제외되어도 무방합니다.

목적 코드나 실행물을 지정한 장소로부터 복제해 갈 수 있게 하는 방식으로 배포할 경우, 동일한 장소로부터 원시 코드를 복제할 수 있는 동등한 접근 방법을 제공한다면 이는 원시 코드를 목적 코드와 함께 복제되도록 설정하지 않았다고 하더라도 원시 코드를 배포하는 것으로 간주됩니다.

제 4 조. 본 허가서에 의해 명시적으로 이루어 지지 않는 한 프로그램에 대한 복제와 개작 및 하위 허가권 설정과 배포가 성립될 수 없습니다. 이와 관련된 어떠한 행위도 무효이며 본 허가서가 보장한 권리는 자동으로 소멸됩니다. 그러나 본 허가서의 규정에 따라 프로그램의 복제물이나 권리를 양도받았던 제3자는 본 허가서의 규정들을 준수하는 한, 배포자의 권리 소멸에 관계없이 사용상의 권리를 계속해서 유지할 수 있습니다.

제 5 조. 본 허가서는 서명이나 날인이 수반되는 형식을 갖고 있지 않기 때문에 피양도자가 본 허가서의 내용을 반드시 받아들여야 할 필요는 없습니다. 그러나 프로그램이나 프로그램에 기반한 2차적 프로그램에 대한 개작 및 배포를 허용하는 것은 본 허가서에 의해서만 가능합니다. 만약 본 허가서에 동의하지 않을 경우에는 이러한 행위들이 법률적으로 금지됩니다. 따라서 프로그램(또는 프로그램에 기반한 2차적 프로그램)을 개작하거나 배포하는 행위는 이에 따른 본 허가서의 내용에 동의한다는 것을 의미하며, 복제와 개작 및 배포에 관한 본 허가서의 조건과 규정들을 모두 받아들이겠다는 의미로 간주됩니다.

제 6 조. 피양도자에 의해서 프로그램(또는 프로그램에 기반한 2차적 프로그램)이 반복적으로 재배포될 경우, 각 단계에서의 피양도자는 본 허가서의 규정에 따른 프로그램의 복제와 개작 및 배포에 대한 권리를 최초의 양도자로부터 양도받은 것으로 자동적으로 간주됩니다. 프로그램(또는 프로그램에 기반한 2차적 프로그램)을 배포할 때는 피양도자의 권리의 행사를 제한할 수 있는 어떠한 사항도 추가할 수 없습니다. 그러나 피양도자에게, 재배포가 일어날 시점에서의 제3의 피양도자에게 본 허가서를 준수하도록 강제할 책임은 부과되지 않습니다.

제 7 조. 법원의 판결이나 특허권 침해에 대한 주장 또는 특허 문제에 국한되지 않은 그밖의 이유들로 인해서 본 허가서의 규정에 배치되는 사항이 발생한다 하더라도 그러한 사항이 선행하거나 본 허가서의 조건과 규정들이 면제되는 것은 아닙니다. 따라서 법원의 명령이나 합의 등에 의해서 본 허가서에 위배되는 사항들이 발생한 상황이라도 양측 모두를 만족시킬 수 없다면 프로그램은 배포될 수 없습니다. 예를 들면, 특정한 특허 관련 허가가 프로그램의 복제물을 직접 또는 간접적인 방법으로 양도받은 임의의 제3자에게

해당 프로그램을 무상으로 재배포할 수 있게 허용하지 않는다면, 그러한 허가와 본 사용 허가를 동시에 만족시키면서 프로그램을 배포할 수 있는 방법은 없습니다.

본 조항은 특정한 상황에서 본 조항의 일부가 유효하지 않거나 적용될 수 없을 경우에도 본 조항의 나머지 부분들을 적용하기 위한 의도로 만들어 졌습니다. 따라서 그 이외의 상황에서는 본 조항을 전체적으로 적용하면 됩니다.

본 조항의 목적은 특허나 저작권 침해 등의 행위를 조장하거나 해당 권리를 인정하지 않으려는 것이 아니라, GPL을 통해서 구현되어 있는 자유 소프트웨어의 배포 체계를 통합적으로 보호하기 위한 것입니다. 많은 사람들이 배포 체계에 대한 신뢰있는 지원을 계속해 줌으로써 소프트웨어의 다양한 분야에 많은 공헌을 해 주었습니다. 소프트웨어를 어떠한 배포 체계로배포할 것인가를 결정하는 것은 전적으로 저작자와 기증자들의 의지에 달려있는 것이지, 일반 사용자들이 강요할 수 있는 문제는 아닙니다.

본 조항은 본 허가서의 다른 조항들에서 무엇이 중요하게 고려되어야 하는 지를 명확하게 설명하기 위한 목적으로 만들어진 것입니다.

제 8 조. 특허나 저작권이 설정된 인터페이스로 인해서 특정 국가에서 프로그램의 배포와 사용이 함께 또는 개별적으로 제한되어 있는 경우, 본 사용 허가서를 프로그램에 적용한 최초의 저작권자는 문제가 발생하지 않는 국가에 한해서 프로그램을 배포한다는 배포상의 지역적 제한 조건을 명시적으로 설정할 수 있으며, 이러한 사항은 본 허가서의 일부로 간주됩니다.

제 9 조. 자유 소프트웨어 재단은 때때로 본 사용 허가서의 개정판이나 신판을 공표할 수 있습니다. 새롭게 공표될 판은 당면한 문제나 현안을 처리하기 위해서 세부적인 내용에 차이가 발생할 수 있지만, 그 근본 정신에는 변함이 없을 것입니다. 각각의 판들은 판번호를 사용해서 구별됩니다. 특정한 판번호와 그 이후 판을 따른다는 사항이 명시된 프로그램에는 해당 판이나 그 이후에 발행된 어떠한 판을 선택해서 적용해도 무방하고, 판번호를 명시하고 있지 않은 경우에는 자유 소프트웨어 재단이 공표한 어떠한 판번호의 판을 적용해도 무방합니다.

제 10 조. 프로그램의 일부를 본 허가서와 배포 기준이 다른 자유 프로그램과 함께 결합하고자 할 경우에는 해당 프로그램의 저작자로부터 서면 승인을 받아야 합니다. 자유 소프트웨어 재단이 저작권을 갖고 있는 소프트웨어의 경우에는 자유 소프트웨어 재단의 승인을 얻어야 합니다. 우리는 이러한 요청을 수락하기 위해서 때때로 예외 기준을 만들기도 합니다. 자유 소프트웨어 재단은 일반적으로 자유 소프트웨어의 2차적 저작물들을 모두 자유로운 상태로 유지시키려는 목적과 소프트웨어의 공유와 재활용을 증진시키려는 두가지 목적을 기준으로 승인 여부를 결정할 것입니다.

보증의 결여

제 11 조. 본 허가서를 따르는 프로그램은 무상으로 양도되기 때문에 관련 법률이 허용하는 한도 내에서 어떠한 형태의 보증도 제공되지 않습니다. 프로그램의 저작권자와 배포자가 공동 또는 개별적으로 별도의 보증을 서면으로 제공할 때를 제외하면, 특정한 목적에 대한 프로그램의 적합성이나 상업성 여부에 대한 보증을 포함한 어떠한 형태의 보증도 명시적이나 묵시적으로 설정되지 않은 “있는 그대로의” 상태로 이 프로그램을 배포합니다. 프로그램과 프로그램의 실행에 따라 발생할 수 있는 모든 위험은 피양도자에게 인수되며 이에 따른 보수 및 복구를 위한 제반 경비 또한 피양도자가 모두 부담해야 합니다.

제 12 조. 저작권자나 배포자가 프로그램의 손상 가능성을 사전에 알고 있었다 하더라도 발생된 손실이 관련 법규에 의해 보호되고 있거나 이에 대한 별도의 서면 보증이설정된 경우가 아니라면, 저작권자나 프로그램을 원래의 상태 또는 개작한 상태로 제공한 배포자는 프로그램의 사용이나 비작동으로 인해 발생된 손실이나 프로그램 자체의 손실에 대해 책임지지 않습니다. 이러한 면책조건은 사용자나 제3자가 프로그램을 조작함으로써 발생된 손실이나 다른 소프트웨어와 프로그램을 함께 동작시키는 것으로 인해서 발생된 데이터의 상실 및 부정확한 산출 결과에만 국한되는 것이 아닙니다. 발생된 손실의 일반성이나 특수성 뿐 아니라 원인의 우발성 및 필연성도 전혀 고려되지 않습니다.

복제와 개작 및 배포에 관한 조건과 규정의 끝

F.3 새로운 프로그램에 GPL을 적용하는 방법

새로운 프로그램을 개발하고 그 프로그램이 많은 사람들에게 최대한 유용하게 사용되기를 원한다면, 본 허가서의 규정에 따라 누구나 자유롭게 개작하고 재배포할 수 있는 자유 소프트웨어로 만드는 것이 최선의 방법입니다.

프로그램을 자유 소프트웨어로 만들기 위해서는 다음과 같은 사항을 프로그램에 추가하면 됩니다. 프로그램에 대한 보증이 제공되지 않는다는 사실을 가장 효과적으로 전달할 수 있는 방법은 원시 코드 파일의 시작 부분에 이러한 사항을 추가하는 것입니다. 각각의 파일에는 최소한 "저작권"을 명시한 행과 본 사용 허가서의 전체 내용을 참고할 수 있는 위치 정보를 명시해야 합니다.

```
프로그램의 이름과 용도를 한 줄 정도로 설명합니다.
Copyright (C) 연도 프로그램 저작자의 이름

이 프로그램은 자유 소프트웨어입니다. 소프트웨어의 피양도자는 자유
소프트웨어 재단이 공표한 GNU 일반 공중 사용 허가서 2판 또는 그
이후 판을 임의로 선택해서, 그 규정에 따라 프로그램을 개작하거나
재배포할 수 있습니다.

이 프로그램은 유용하게 사용될 수 있으리라는 희망에서 배포되고
있지만, 특정한 목적에 맞는 적합성 여부나 판매용으로 사용할 수
있으리라는 묵시적인 보증을 포함한 어떠한 형태의 보증도 제공하지
않습니다. 보다 자세한 사항에 대해서는 GNU 일반 공중 사용 허가서를
참고하시기 바랍니다.

GNU 일반 공중 사용 허가서는 이 프로그램과 함께 제공됩니다. 만약,
이 문서가 누락되어 있다면 자유 소프트웨어 재단으로 문의하시기
바랍니다. (자유 소프트웨어 재단: Free Software Foundation, Inc.,
59 Temple Place - Suite 330, Boston, MA 02111-1307, USA)
```

또한, 사용자들이 프로그램을 배포한 사람에게 전자 메일과 서면으로 연락할 수 있는 정보를 추가해야 합니다.

프로그램이 명령어 입력 방식에 의한 대화형 구조를 택하고 있다면, 프로그램이 대화형 방식으로 실행되었을 때 다음과 같은 주의 사항이 출력되어야 합니다.

```
Gnomovision version 69, Copyright (C) 연도 프로그램 저작자의 이름
Gnomovision 프로그램에는 제품에 대한 어떠한 형태의 보증도
제공되지 않습니다. 보다 자세한 사항은 show w 명령어를
실행해서 참고할 수 있습니다. 이 프로그램은 자유 소프트웨어입니다.
이 프로그램은 배포 규정을 만족시키는 조건하에서 자유롭게
재배포될 수 있습니다. 배포에 대한 규정들은 show c 명령어를
통해서 참고할 수 있습니다.
```

"show w"와 "show c"는 GPL의 해당 부분을 출력하기 위한 가상의 명령어입니다. 따라서 "show w"나 "show c"가 아닌 다른 형태를 사용해도 무방하며, 마우스 클릭이나 메뉴 방식과 같은 프로그램에 적합한 다른 형식을 사용해도 괜찮습니다.

만약, 프로그램 저작자가 학교나 기업과 같은 단체나 기관에 고용되어 있다면 프로그램의 자유로운 배포를 위해서 고용주나 해당 기관장으로부터 프로그램에 대한 "저작권 포기 각서"를 받아야 합니다. 예를 들면 다음과 같은 형식이 될 수 있다. (아래의 문구를 실제로 사용할 경우에는 예로 사용된 이름들을 실제 이름으로 대체하면 됩니다.)

```
본사는 제임스 해커가 만든 (컴파일러에서 패스를 생성하는)
Gnomovision 프로그램에 관련된 모든 저작권을 포기합니다.

1989년 4월 1일 부사장: Ty Coon Ty Coon의 서명
```

GNU 일반 공중 사용 허가서는 자유 소프트웨어를 독점 소프트웨어와 함께 결합시키는 것을 허용하지 않습니다. 만약, 작성된 프로그램이 서브루틴 라이브러리일 경우에는 독점 소프트웨어가 해당 라이브러리를 링크할 수 있도록 허용하는 것이 보다 효과적으로 활용될 수 있는 방법이라고 생각할 수도 있을 것입니다. 이러한 경우에는 본 허가서 GNU Lesser General Public License를 사용함으로써 소기의 목적을 충족시킬 수 있습니다.