Red Hat Enterprise Linux 8

고급 RHEL 8 설치 수행

Kickstart를 사용하여 RHEL 8 설치

Last Updated: 2022-09-04

Red Hat Enterprise Linux 8 고급 RHEL 8 설치 수행

Kickstart를 사용하여 RHEL 8 설치

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

법적 공지

Copyright © 2022 | You need to change the HOLDER entity in the en-US/Performing_an_advanced_RHEL_8_installation.ent file |.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

초록

이 문서는 Kickstart를 사용하여 고급 Red Hat Enterprise Linux 설치를 수행하고 고급 설치 옵션을 구성하려는 사용자를 위한 것입니다.

차례

보다 포괄적 수용을 위한 오픈 소스 용어 교체

Red Hat은 코드, 문서, 웹 속성에서 문제가 있는 용어를 교체하기 위해 최선을 다하고 있습니다. 먼저 마스터(master), 슬레이브(slave), 블랙리스트(blacklist), 화이트리스트(whitelist) 등 네 가지 용어를 교체하고 있습니다. 이러한 변경 작업은 작업 범위가 크므로 향후 여러 릴리스에 걸쳐 점차 구현할 예정입니다. 자세한 내용은 CTO Chris Wright의 메시지 를 참조하십시오.

RED HAT 문서에 관한 피드백 제공

문서에 대한 피드백에 감사드립니다. 어떻게 개선할 수 있는지 알려주십시오.

특정 문구에 대한 의견 제출

1. **Multi-page HTML** 형식으로 설명서를 보고 페이지가 완전히 로드된 후 오른쪽 상단 모서리에 **피드백** 버튼이 표시되는지 확인합니다.
2. 커서를 사용하여 주석 처리할 텍스트 부분을 강조 표시합니다.
3. 강조 표시된 텍스트 옆에 표시되는 **피드백 추가** 버튼을 클릭합니다.
4. 의견을 추가하고 **제출** 을 클릭합니다.

Bugzilla를 통해 피드백 제출(등록 필요)

1. Bugzilla 웹 사이트에 로그인합니다.
2. **버전** 메뉴에서 올바른 버전을 선택합니다.
3. **Summary** (요약) 필드에 설명 제목을 입력합니다.
4. **Description** (설명) 필드에 개선을 위한 제안을 입력합니다. 문서의 관련 부분에 대한 링크를 포함합니다.
5. **버그 제출**을 클릭합니다.

1장. 소개

Red Hat Enterprise Linux 8은 적은 노력으로 보다 빠르게 워크로드를 제공하는 데 필요한 툴을 사용하여 하이브리드 클라우드 배포 전반에 걸쳐 안정적이고 안전하며 일관된 기반을 제공합니다. 지원되는 하이퍼바이저 및 클라우드 공급자 환경에 게스트로 배포할 수 있을 뿐 아니라 물리적 인프라에 배포할 수 있기 때문에 애플리케이션이 선도적인 하드웨어 아키텍처 플랫폼의 혁신을 활용할 수 있습니다.

1.1. 지원되는 아키텍처

Red Hat Enterprise Linux는 다음과 같은 아키텍처를 지원합니다.

- AMD, Intel 및 ARM 64비트 아키텍처
- IBM Power Systems, Little Endian
  - IBM Power System LC 서버
  - IBM Power System AC 서버
  - IBM Power System L 서버
- 64-bit IBM Z

참고

IBM Power Servers에 대한 설치 지침은 *IBM 설치 설명서를 참조하십시오*. 시스템이 RHEL 설치를 위해 지원되는지 확인하려면 https://catalog.redhat.com 및 https://access.redhat.com/articles/rhel-limits 을 참조하십시오.

1.2. 설치 용어

이 섹션에서는 Red Hat Enterprise Linux 설치 용어에 대해 설명합니다. 업스트림 또는 다운스트림에 따라 서로 다른 용어를 동일한 개념에 사용할 수 있습니다.

Anaconda: Fedora, Red Hat Enterprise Linux 및 파생 제품에서 사용되는 운영 체제 설치 프로그램입니다. Anaconda는 C로 작성된 Gtk 위젯(C), systemd 유닛 및 dracut 라이브러리와 같은 추가 파일이 있는 Python 모듈 및 스크립트 세트입니다. 이를 통해 사용자가 결과(대상) 시스템의 매개 변수를 설정할 수 있는 툴을 형성합니다. 이 문서에서 **설치 프로그램**이라는 용어는 **Anaconda**의 설치 측면을 나타냅니다.

2장. 설치 방법

요구 사항에 따라 여러 가지 방법으로 Red Hat Enterprise Linux를 설치할 수 있습니다. 다음 섹션을 검토하여 요구 사항에 가장 적합한 설치 방법을 확인합니다.

2.1. 사용 가능한 설치 방법

다음 방법을 사용하여 Red Hat Enterprise Linux를 설치할 수 있습니다.

- GUI 기반 설치
- 시스템 또는 클라우드 이미지 기반 설치
- 고급 설치

참고

이 문서에서는 고급 설치 방법을 사용하여 Red Hat Enterprise Linux를 설치하는 방법에 대해 자세히 설명합니다.

고급 설치

다음과 같은 고급 설치 방법을 사용할 수 있습니다.

- **Kickstart를 사용하여 자동화된 RHEL 설치 수행: Kickstart를** 사용하여 Red Hat Enterprise Linux 설치. Kickstart는 무인 운영 체제 설치 작업을 실행할 수 있는 자동화된 설치입니다.
- CDN(**Content Delivery Network)에서 RHEL 등록 및 설치:**CDN(Content Delivery Network)의 모든 아키텍처에 Red Hat Enterprise Linux 등록 및 설치. CDN에서 설치 패키지를 다운로드 및 설치하기 전에 등록이 수행됩니다. 이 설치 방법은 그래픽 사용자 인터페이스와 Kickstart에서 지원합니다.
- **VNC를 사용하여 원격 RHEL 설치를 수행합니다.** RHEL 설치 프로그램은 두 개의 VNC 설치 모드인 Direct 및 Connect를 제공합니다. 연결이 설정되면 두 모드가 다릅니다. 선택한 모드는 환경에 따라 다릅니다.
- **PXE를 사용하여 네트워크에서 RHEL 설치 :**네트워크 설치를 통해 Red Hat Enterprise Linux를 설치 서버에 액세스할 수 있는 시스템에 설치할 수 있습니다. 최소한 네트워크 설치에는 두 개의 시스템이 필요합니다.

시스템 또는 클라우드 이미지 기반 설치

가상 및 클라우드 환경에서만 시스템 또는 클라우드 이미지 기반 설치 방법을 사용할 수 있습니다. 시스템 또는 클라우드 이미지 기반 설치를 수행하려면 Red Hat Image Builder를 사용합니다. Image Builder는 클라우드 배포를 위한 시스템 이미지를 포함하여 Red Hat Enterprise Linux의 사용자 지정 시스템 이미지를 생성합니다.

이미지 빌더를 사용하여 RHEL을 설치하는 방법에 대한 자세한 내용은 *사용자 지정된 RHEL 시스템 이미지 구성* 문서를 참조하십시오.

GUI 기반 설치

다음과 같은 GUI 기반 설치 방법을 사용할 수 있습니다.

- 고객 **포털에서 ISO 이미지를 사용하여 RHEL 설치: 고객 포털에서 DVD ISO** 이미지 파일을 다운로드하여 Red Hat Enterprise Linux 설치. 설치가 완료된 후 등록이 수행됩니다. 이 설치 방법은 GUI와 Kickstart에서 지원합니다.

- **Content Delivery Network에서 RHEL 등록 및 설치:**시스템을 등록하고 서브스크립션을 연결하며 CDN(Content Delivery Network)에서 Red Hat Enterprise Linux를 설치합니다. 이 설치 방법은 **부팅 ISO** 및 **DVD ISO** 이미지 파일에서 지원합니다. 그러나 **부팅 ISO** 이미지 파일의 기본 설치 소스로 부팅 ISO 이미지 파일을 사용하는 것이 좋습니다. CDN에서 설치 패키지를 다운로드 및 설치하기 전에 등록이 수행됩니다. 이 설치 방법은 GUI와 Kickstart에서 지원합니다.

추가 리소스

- *표준 RHEL 8 설치 수행*

I 부. KICKSTART를 사용하여 자동 설치 수행

3장. KICKSTART 설치 기본 사항

다음은 Kickstart에 대한 기본 정보와 이 정보를 사용하여 Red Hat Enterprise Linux 설치를 자동화하는 방법을 제공합니다.

3.1. KICKSTART 설치란

Kickstart는 RHEL 설치 프로세스를 부분적으로 또는 완전히 자동화하는 방법을 제공합니다.

Kickstart 파일에는 일부 또는 전체 RHEL 설치 옵션이 포함되어 있습니다. 예를 들어 시간대, 드라이브 파티셔닝 방법 또는 설치해야 하는 패키지를 설치할 수 있습니다. 준비된 Kickstart 파일을 제공하면 사용자의 개입 없이도 설치할 수 있습니다. 이는 Red Hat Enterprise Linux를 한 번에 다수의 시스템에 배포할 때 특히 유용합니다.

Kickstart 파일은 소프트웨어 선택과 관련된 추가 옵션도 제공합니다. 그래픽 설치 인터페이스를 사용하여 Red Hat Enterprise Linux를 수동으로 설치하는 경우 소프트웨어 선택은 사전 정의된 환경 및 애드온으로 제한됩니다. Kickstart 파일을 사용하면 개별 패키지를 설치하거나 제거할 수도 있습니다.

Kickstart 파일은 단일 서버 시스템에 보관하고 설치 중에 개별 컴퓨터에서 읽을 수 있습니다. 이 설치 방법은 단일 Kickstart 파일을 사용하여 여러 시스템에 Red Hat Enterprise Linux를 설치할 수 있도록 지원하므로 네트워크 및 시스템 관리자에게 이상적입니다.

모든 Kickstart 스크립트와 실행 로그 파일은 설치 문제를 디버깅하는 데 도움이 되도록 새로 설치된 시스템의 **/tmp** 디렉토리에 저장됩니다. Anaconda에서 생성된 출력 킥스타트는 설치에 사용되는 킥스타트 및 대상 시스템의 [filename]/root에 저장되며 kickstart 스크립트릿 실행의 로그는 **/var/log/anaconda**에 저장됩니다.

참고

이전 버전의 Red Hat Enterprise Linux에서는 시스템을 업그레이드하는 데 Kickstart를 사용할 수 있었습니다. Red Hat Enterprise Linux 7부터는 이 기능이 제거되었으며 시스템 업그레이드는 특수 툴에 의해 처리됩니다. Red Hat Enterprise Linux 8으로의 업그레이드에 대한 자세한 내용은 RHEL 7에서 RHEL 8로의 업그레이드 및 RHEL 채택시 고려 사항을 참조하십시오.

3.2. 자동 설치 워크플로

Kickstart 설치는 로컬 DVD, 로컬 하드 드라이브 또는 NFS, FTP, HTTP 또는 HTTPS 서버를 사용하여 수행할 수 있습니다. 이 섹션에서는 Kickstart 사용에 대한 간략한 개요를 제공합니다.

1. Kickstart 파일을 만듭니다. 직접 작성하고 수동 설치 후 저장된 Kickstart 파일을 복사하거나 온라인 생성 도구를 사용하여 파일을 생성하고 나중에 편집할 수 있습니다. Kickstart 파일 생성을 참조하십시오.

2. HTTP(S), FTP 또는 NFS 서버를 사용하여 이동식 미디어, 하드 드라이브 또는 네트워크 위치의 설치 프로그램에서 Kickstart 파일을 사용하도록 설정합니다. 설치 프로그램에서 Kickstart 파일을 사용할 수 있도록 만들기를 참조하십시오.

3. 설치를 시작하는 데 사용할 부팅 매체를 만듭니다. 부팅 가능 설치 매체 생성 및 PXE를 사용하여 네트워크에서 설치 준비를 참조하십시오.

4. 설치 프로그램에서 설치 소스를 사용할 수 있도록 합니다. Kickstart 설치를 위한 설치 소스 생성을 참조하십시오.

5. 부팅 미디어와 Kickstart 파일을 사용하여 설치를 시작합니다. Kickstart 설치 시작을 참조하십시오.

Kickstart 파일에 필수 명령 및 섹션이 포함된 경우 설치가 자동으로 완료됩니다. 이러한 필수 부품 중 하나 이상이 누락되었거나 오류가 발생한 경우 설치를 완료하려면 수동 개입이 필요합니다.

참고

UEFI Secure Boot가 활성화된 시스템에 Red Hat Enterprise Linux 베타 릴리스를 설치하려면 먼저 UEFI Secure Boot 옵션을 비활성화한 다음 설치를 시작합니다.

UEFI Secure Boot를 사용하려면 시스템의 펌웨어가 해당 공개 키를 사용하여 확인하는 인식된 개인 키로 운영 체제 커널에 서명해야 합니다. Red Hat Enterprise Linux 베타 릴리스의 경우 커널은 Red Hat 베타 특정 개인 키와 서명되어 시스템이 기본적으로 인식되지 못합니다. 결과적으로 시스템이 설치 미디어를 부팅하지 못합니다.

4장. KICKSTART 파일 생성

다음 방법을 사용하여 Kickstart 파일을 생성할 수 있습니다.

- 온라인 Kickstart 구성 툴을 사용합니다.
- 수동 설치에서 생성된 Kickstart 파일을 복사합니다.
- 전체 Kickstart 파일을 수동으로 작성합니다.
- Red Hat Enterprise Linux 8 설치를 위한 Red Hat Enterprise Linux 7 Kickstart 파일을 변환합니다. 변환 도구에 대한 자세한 내용은 Kickstart 생성기 랩 을 참조하십시오.
- 가상 및 클라우드 환경의 경우 Image Builder를 사용하여 사용자 지정 시스템 이미지를 만듭니다.

Kickstart 파일을 수동으로 편집하는 경우에만 몇 가지 특정 설치 옵션을 구성할 수 있습니다.

4.1. KICKSTART 구성 툴로 KICKSTART 파일 생성

Red Hat 고객 포털 계정이 있는 사용자는 고객 포털 랩의 Kickstart 생성 툴을 사용하여 온라인에서 Kickstart 파일을 생성할 수 있습니다. 이 툴은 기본 구성을 안내하고 결과 Kickstart 파일을 다운로드할 수 있도록 합니다.

참고

현재 툴은 고급 파티션을 지원하지 않습니다.

사전 요구 사항

- Red Hat 고객 포털 계정과 유효한 Red Hat 서브스크립션이 있어야 합니다.

절차

1. https://access.redhat.com/labsinfo/kickstartconfig에서 Kickstart 생성기 랩 정보 페이지를 엽니다.
2. 제목 왼쪽에 있는 **Go to Application** 버튼을 클릭하고 다음 페이지가 로드될 때까지 기다립니다.
3. 드롭다운 메뉴에서 **Red Hat Enterprise Linux 8**을 선택하고 페이지가 업데이트될 때까지 기다립니다.
4. 양식의 필드를 사용하여 설치할 시스템을 설명합니다.
   양식 왼쪽의 링크를 사용하여 양식의 섹션 사이를 빠르게 탐색할 수 있습니다.
5. 생성된 Kickstart 파일을 다운로드하려면 페이지 상단에 있는 빨간색 **다운로드** 버튼을 클릭합니다.
   웹 브라우저에서 파일을 저장합니다.

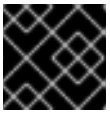
4.2. 수동 설치를 수행하여 KICKSTART 파일 생성

Kickstart 파일 생성에 권장되는 방법은 Red Hat Enterprise Linux의 수동 설치로 생성된 파일을 사용하는 것입니다. 설치가 완료되면 설치 중에 수행한 모든 선택 사항이 설치된 시스템의 **/root/** 디렉터리에 있는 **anaconda-ks.cfg** 라는 Kickstart 파일에 저장됩니다. 이 파일을 사용하여 이전과 동일한 방식으로 설치를

재현할 수 있습니다. 또는 이 파일을 복사하여 필요한 사항을 변경하고 추가 설치를 위해 결과 구성 파일을 사용합니다.

절차

1. RHEL을 설치합니다. 자세한 내용은 표준 RHEL 8 설치 수행을 참조하십시오.
   설치 중에 관리자 권한이 있는 사용자를 만듭니다.
2. 설치를 완료하고 설치된 시스템으로 재부팅합니다.
3. 관리자 계정으로 시스템에 로그인합니다.
4. **/root/anaconda-ks.cfg** 파일을 선택한 위치로 복사합니다.



중요

파일에는 사용자와 암호에 대한 정보가 포함되어 있습니다.

- 터미널에서 파일 내용을 표시하려면 다음을 수행합니다.

```
# cat /root/anaconda-ks.cfg
```

  출력을 복사하여 선택한 다른 파일에 저장할 수 있습니다.

- 파일을 다른 위치로 복사하려면 파일 관리자를 사용합니다. 루트가 아닌 사용자가 파일을 읽을 수 있도록 복사에서 권한을 변경해야 합니다.

추가 리소스

- 표준 RHEL 8 설치 수행

4.3. 이전 RHEL 설치에서 KICKSTART 파일 변환

Kickstart Converter 툴을 사용하여 RHEL 8 또는 9 설치에서 사용할 RHEL 7 Kickstart 파일을 변환하거나 RHEL 9에서 사용할 RHEL 8 Kickstart 파일을 변환할 수 있습니다. 툴 및 이를 사용하여 RHEL Kickstart 파일을 변환하는 방법에 대한 자세한 내용은 https://access.redhat.com/labs/kickstartconvert/을 참조하십시오.

4.4. 이미지 빌더를 사용하여 사용자 정의 이미지 생성

Red Hat Image Builder를 사용하여 가상 및 클라우드 배포를 위한 사용자 지정 시스템 이미지를 만들 수 있습니다.

Image Builder를 사용하여 사용자 지정된 이미지 생성에 대한 자세한 내용은 *사용자 정의 RHEL 시스템 이미지* 문서를 참조하십시오.

5장. 설치 프로그램에서 KICKSTART 파일을 사용할 수 있도록 설정

다음은 대상 시스템에서 설치 프로그램에서 Kickstart 파일을 사용할 수 있도록 하는 방법에 대한 정보를 제공합니다.

5.1. 네트워크 기반 설치를 위한 포트

다음 표에는 각 네트워크 기반 설치 유형에 대한 파일을 제공하는 서버에서 열어야 하는 포트가 나열되어 있습니다.

표 5.1. 네트워크 기반 설치를 위한 포트

사용된 프로토콜	오픈할 포트
HTTP	80
HTTPS	443
FTP	21
NFS	2049, 111, 20048
TFTP	69

추가 리소스

- 네트워크 보안

5.2. NFS 서버에서 KICKSTART 파일을 사용할 수 있도록 설정

다음 절차에서는 Kickstart 스크립트 파일을 NFS 서버에 저장하는 방법을 설명합니다. 이 방법을 사용하면 Kickstart 파일에 물리적 미디어를 사용하지 않고도 단일 소스에서 여러 시스템을 설치할 수 있습니다.

사전 요구 사항

- 로컬 네트워크에 Red Hat Enterprise Linux 8이 있는 서버에 대한 관리자 수준의 액세스 권한이 있어야 합니다.
- 설치할 시스템은 서버에 연결할 수 있어야 합니다.
- 서버의 방화벽은 설치하려는 시스템에서 연결을 허용해야 합니다. 자세한 내용은 네트워크 기반 설치 용 포트를 참조하십시오.

절차

1. root로 다음 명령을 실행하여 **nfs-utils** 패키지를 설치합니다.

```
# yum install nfs-utils
```

2. Kickstart 파일을 NFS 서버의 디렉터리에 복사합니다.

3. 텍스트 편집기를 사용하여 **/etc/exports** 파일을 열고 다음 구문으로 행을 추가합니다.

```
/exported_directory/ clients
```

4. */exported_directory/*를 Kickstart 파일이 포함된 디렉토리의 전체 경로로 바꿉니다. *클라이언트* 대신 이 NFS 서버에서 설치할 컴퓨터의 호스트 이름 또는 IP 주소를 사용합니다. 모든 컴퓨터가 ISO 이미지를 사용하도록 하려면 모든 컴퓨터가 ISO 이미지에 액세스할 수 있는 서브네트워크 또는 별표 기호(*)를 사용합니다. 이 필드의 형식에 대한 자세한 내용은 *exports(5)* 도움말 페이지를 참조하십시오.
/rhel8-install/ 디렉토리를 모든 클라이언트에 읽기 전용으로 사용할 수 있도록 하는 기본 구성은 다음과 같습니다.

```
/rhel8-install *
```

5. **/etc/exports** 파일을 저장하고 텍스트 편집기를 종료합니다.

6. nfs 서비스를 시작합니다.

```
# systemctl start nfs-server.service
```

/etc/exports 파일을 변경하기 전에 서비스가 실행 중인 경우 실행 중인 NFS 서버가 구성을 다시 로드하기 위해 다음 명령을 입력합니다.

```
# systemctl reload nfs-server.service
```

이제 NFS를 통해 Kickstart 파일에 액세스할 수 있으며 설치에 사용할 준비가 되었습니다.

참고

Kickstart 소스를 지정할 때 **nfs:** 를 프로토콜로, 서버의 호스트 이름 또는 IP 주소, 콜론 기호(**:**), 파일을 보유한 디렉터리 내부의 경로를 사용합니다. 예를 들어 서버의 호스트 이름이 **myserver.example.com**이고 **/rhel8-install/my-ks.cfg**에 파일을 저장한 경우 **inst.ks=nfs:myserver.example.com:/rhel8-install/my-ks.cfg**를 설치 소스 부팅 옵션으로 지정합니다.

추가 리소스

- PXE를 사용하여 네트워크에서 설치 준비

5.3. HTTP 또는 HTTPS 서버에서 KICKSTART 파일을 사용할 수 있도록 설정

다음 절차에서는 HTTP 또는 HTTPS 서버에 Kickstart 스크립트 파일을 저장하는 방법을 설명합니다. 이 방법을 사용하면 Kickstart 파일에 물리적 미디어를 사용하지 않고도 단일 소스에서 여러 시스템을 설치할 수 있습니다.

사전 요구 사항

- 로컬 네트워크에 Red Hat Enterprise Linux 8이 있는 서버에 대한 관리자 수준의 액세스 권한이 있어야 합니다.

- 설치할 시스템은 서버에 연결할 수 있어야 합니다.

- 서버의 방화벽은 설치하려는 시스템에서 연결을 허용해야 합니다. 자세한 내용은 네트워크 기반 설치 용 포트를 참조하십시오.

절차

1. Kickstart 파일을 HTTP에 저장하려면 **httpd** 패키지를 설치합니다.

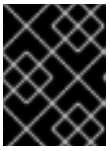
```
# yum install httpd
```

Kickstart 파일을 HTTPS에 저장하려면 **httpd** 및 **mod_ssl** 패키지를 설치합니다.

```
# yum install httpd mod_ssl
```

주의

Apache 웹 서버 구성이 SSL 보안을 활성화하는 경우 TLSv1 프로토콜만 활성화하고 SSLv2 및 SSLv3을 비활성화하는지 확인합니다. 이는 POODLE SSL 취약점(CVE-2014-3566) 때문입니다. 자세한 내용은 https://access.redhat.com/solutions/1232413을 참조하십시오.



중요

자체 서명된 인증서가 있는 HTTPS 서버를 사용하는 경우 **inst.noverifyssl** 옵션을 사용하여 설치 프로그램을 부팅해야 합니다.

2. Kickstart 파일을 HTTP(S) 서버에 **/var/www/html/** 디렉터리의 하위 디렉터리로 복사합니다.
3. httpd 서비스를 시작합니다.

```
# systemctl start httpd.service
```

이제 Kickstart 파일에 액세스할 수 있으며 설치에 사용할 준비가 되었습니다.

참고

Kickstart 파일의 위치를 지정하는 경우 **http://** 또는 **https://**를 프로토콜로 서버의 호스트 이름 또는 IP 주소 및 HTTP 서버 루트에 상대적인 Kickstart 파일의 경로를 사용합니다. 예를 들어 HTTP를 사용하는 경우 서버의 호스트 이름은 **myserver.example.com** 이며, Kickstart 파일을 **/var/www/html/rhel8-install/my-ks.cfg** 로 복사한 경우 **http://myserver.example.com/rhel8-install/my-ks.cfg** 을 파일 위치로 지정합니다.

추가 리소스

- 다양한 유형의 서버 배포

5.4. FTP 서버에서 KICKSTART 파일을 사용할 수 있도록 설정

다음 절차에서는 Kickstart 스크립트 파일을 FTP 서버에 저장하는 방법을 설명합니다. 이 방법을 사용하면 Kickstart 파일에 물리적 미디어를 사용하지 않고도 단일 소스에서 여러 시스템을 설치할 수 있습니다.

사전 요구 사항

- 로컬 네트워크에 Red Hat Enterprise Linux 8이 있는 서버에 대한 관리자 수준의 액세스 권한이 있어야 합니다.
- 설치할 시스템은 서버에 연결할 수 있어야 합니다.
- 서버의 방화벽은 설치하려는 시스템에서 연결을 허용해야 합니다. 자세한 내용은 네트워크 기반 설치 용 포트를 참조하십시오.

절차

1. root로 다음 명령을 실행하여 **vsftpd** 패키지를 설치합니다.

```
# yum install vsftpd
```

2. 텍스트 편집기에서 **/etc/vsftpd/vsftpd.conf** 구성 파일을 열고 편집합니다.
   a. **anonymous_enable=NO** 행을 **anonymous_enable=YES**로 변경합니다.
   b. **write_enable=YES** 행을 **write_enable=NO**로 변경합니다.
   c. **pasv_min_port=*min_port*** 및 **pasv_max_port=*max_port*** 행을 추가합니다. *min_port* 및 *max_port* 를 Passive 모드에서 FTP 서버에서 사용하는 포트 번호 범위(예: **10021** 및 **10031)**로 바꿉니다.
   이 단계는 다양한 방화벽/NAT 설정을 갖춘 네트워크 환경에서 필요할 수 있습니다.
   d. 선택적으로 구성에 사용자 지정 변경 사항을 추가합니다. 사용 가능한 옵션은 **vsftpd.conf(5)** 도움말 페이지를 참조하십시오. 이 절차에서는 기본 옵션이 사용된다고 가정합니다.

주의

vsftpd.conf 파일에 SSL/TLS 보안을 구성한 경우 TLSv1 프로토콜만 활성화하고 SSLv2 및 SSLv3을 비활성화해야 합니다. 이는 POODLE SSL 취약점(CVE-2014-3566) 때문입니다. 자세한 내용은 https://access.redhat.com/solutions/1234773을 참조하십시오.

3. 서버 방화벽을 구성합니다.
   a. 방화벽을 활성화합니다.

```
# systemctl enable firewalld
# systemctl start firewalld
```

   b. 방화벽에서 이전 단계의 FTP 포트 및 포트 범위를 활성화합니다.

```
# firewall-cmd --add-port min_port-max_port/tcp --permanent
# firewall-cmd --add-service ftp --permanent
# firewall-cmd --reload
```

*min_port-max_port*를 **/etc/vsftpd/vsftpd.conf** 구성 파일에 입력한 포트 번호로 바꿉니다.

4. Kickstart 파일을 FTP 서버에 **/var/ftp/** 디렉토리 또는 해당 하위 디렉터리에 복사합니다.
5. 파일에 올바른 SELinux 컨텍스트 및 액세스 모드가 설정되어 있는지 확인합니다.

```
# restorecon -r /var/ftp/your-kickstart-file.ks
# chmod 444 /var/ftp/your-kickstart-file.ks
```

6. **vsftpd** 서비스를 시작합니다.

```
# systemctl start vsftpd.service
```

/etc/vsftpd/vsftpd.conf 파일을 변경하기 전에 서비스가 실행 중인 경우 서비스를 다시 시작하여 편집된 파일을 로드합니다.

```
# systemctl restart vsftpd.service
```

부팅 프로세스 중에 시작되도록 **vsftpd** 서비스를 활성화합니다.

```
# systemctl enable vsftpd
```

이제 Kickstart 파일에 액세스할 수 있으며 동일한 네트워크의 시스템에서 설치할 준비가 되었습니다.

참고

설치 소스를 구성할 때 프로토콜, 서버의 호스트 이름 또는 IP 주소, FTP 서버 루트를 기준으로 Kickstart 파일의 경로를 사용하여 **ftp://**를 사용합니다. 예를 들어 서버의 호스트 이름이 **myserver.example.com** 이고 파일을 **/var/ftp/my-ks.cfg** 에 복사한 경우 설치 소스로 **ftp://myserver.example.com/my-ks.cfg** 을 지정합니다.

5.5. 로컬 볼륨에서 KICKSTART 파일을 사용할 수 있도록 설정

다음 절차에서는 설치할 시스템의 볼륨에 Kickstart 스크립트 파일을 저장하는 방법을 설명합니다. 이 방법을 사용하면 다른 시스템의 필요성을 무시할 수 있습니다.

사전 요구 사항

- 설치할 시스템으로 이동할 수 있는 드라이브(예: USB 스틱)가 있어야 합니다.
- 드라이브에는 설치 프로그램에서 읽을 수 있는 파티션이 포함되어야 합니다. 지원되는 유형은 **ext2**,**ext3**,**ext4**,**xfs** 및 **fat** 입니다.
- 드라이브가 이미 시스템 및 마운트된 볼륨에 연결되어 있어야 합니다.

절차

1. 볼륨 정보를 나열하고 Kickstart 파일을 복사할 볼륨의 UUID를 확인합니다.

```
# lsblk -l -p -o name,rm,ro,hotplug,size,type,mountpoint,uuid
```

2. 볼륨의 파일 시스템으로 이동합니다.
3. Kickstart 파일을 이 파일 시스템에 복사합니다.
4. 나중에 **inst.ks=** 옵션과 함께 사용하도록 문자열을 기록합니다. 이 문자열은 다음과 같은 형식으로 되어 있습니다 **:UUID=*volume-UUID*:*path/to/kickstart-file.cfg***. 경로는 파일 시스템 계층 구조의 / 루트가 아닌 파일 시스템 루트와 관련이 있습니다. *volume-UUID* 를 앞에서 언급한 UUID로 바꿉니다.
5. 모든 드라이브 볼륨을 마운트 해제합니다.

```
# umount /dev/xyz ...
```

모든 볼륨을 명령에 공백으로 구분하여 추가합니다.

5.6. 자동 로드를 위해 로컬 볼륨에서 KICKSTART 파일을 사용하도록 설정

특별히 이름이 지정된 Kickstart 파일인 는 설치할 시스템에서 특별히 이름이 지정된 볼륨의 루트에 있을 수 있습니다. 이를 통해 다른 시스템의 요구 사항을 무시하고 설치 프로그램에서 파일을 자동으로 로드할 수 있습니다.

사전 요구 사항

- 설치할 시스템으로 이동할 수 있는 드라이브(예: USB 스틱)가 있어야 합니다.
- 드라이브에는 설치 프로그램에서 읽을 수 있는 파티션이 포함되어야 합니다. 지원되는 유형은 **ext2**,**ext3**,**ext4**,**xfs** 및 **fat** 입니다.
- 드라이브가 이미 시스템 및 마운트된 볼륨에 연결되어 있어야 합니다.

절차

1. Kickstart 파일을 복사할 볼륨 정보를 나열합니다.

```
# lsblk -l -p
```

2. 볼륨의 파일 시스템으로 이동합니다.
3. Kickstart 파일을 이 파일 시스템의 루트로 복사합니다.
4. Kickstart 파일의 이름을 **ks.cfg**로 바꿉니다.
5. 볼륨의 이름을 **OEMDRV**로 변경합니다.
   - **ext2**,**ext3** 및 **ext4** 파일 시스템의 경우 다음을 수행합니다.

```
# e2label /dev/xyz OEMDRV
```

   - XFS 파일 시스템의 경우 다음을 수행합니다.

```
# xfs_admin -L OEMDRV /dev/xyz
```

*/dev/xyz*를 볼륨 블록 장치의 경로로 바꿉니다.

6. 모든 드라이브 볼륨을 마운트 해제합니다.

```
# umount /dev/xyz ...
```

모든 볼륨을 명령에 공백으로 구분하여 추가합니다.

6장. KICKSTART 설치를 위한 설치 소스 생성

이 섹션에서는 필수 리포지토리 및 소프트웨어 패키지가 포함된 DVD ISO 이미지를 사용하여 부팅 ISO 이미지에 대한 설치 소스를 생성하는 방법을 설명합니다.

6.1. 설치 소스 유형

최소 부팅 이미지에 다음 설치 소스 중 하나를 사용할 수 있습니다.

- **DVD:** DVD ISO 이미지를 DVD로 확장합니다. DVD는 설치 소스(소프트웨어 패키지 소스)로 자동으로 사용됩니다.
- **하드 드라이브 또는 USB 드라이브:** DVD ISO 이미지를 드라이브에 복사하고 드라이브에서 소프트웨어 패키지를 설치하도록 설치 프로그램을 구성합니다. USB 드라이브를 사용하는 경우 설치를 시작하기 전에 시스템에 연결되어 있는지 확인합니다. 설치 프로그램이 시작된 후에는 설치 프로그램이 미디어를 감지할 수 없습니다.
  - **하드 드라이브 제한**: 하드 드라이브의 DVD ISO 이미지는 설치 프로그램이 마운트할 수 있는 파일 시스템이 있는 파티션에 있어야 합니다. 지원되는 파일 시스템은 **xfs**,**ext2**,**ext3**,**ext4** 및 **vfat(FAT32)** 입니다.

주의

Microsoft Windows 시스템에서 하드 드라이브를 포맷할 때 사용되는 기본 파일 시스템은 NTFS입니다. exFAT 파일 시스템도 사용할 수 있습니다. 그러나 설치하는 동안 이러한 파일 시스템을 마운트할 수 없습니다. Microsoft Windows에서 하드 드라이브 또는 USB 드라이브를 설치 소스로 생성하는 경우 드라이브를 FAT32로 포맷했는지 확인합니다. FAT32 파일 시스템은 4GiB 보다 큰 파일을 저장할 수 없습니다.

Red Hat Enterprise Linux 8에서는 로컬 하드 드라이브의 디렉터리에서 설치를 활성화할 수 있습니다. 이렇게 하려면 DVD ISO 이미지의 내용을 하드 드라이브의 디렉터리에 복사한 다음, ISO 이미지 대신 설치 소스로 디렉터리를 지정해야 합니다. 예: **inst.repo=hd:<device>:<path to the directory>**

- **네트워크 위치:** DVD ISO 이미지 또는 설치 트리 (DVD ISO 이미지의 추출 내용)를 네트워크 위치에 복사하고 다음 프로토콜을 사용하여 네트워크를 통해 설치를 수행합니다.
  - **NFS:** DVD ISO 이미지는 NFS(네트워크 파일 시스템) 공유에 있습니다.
  - **HTTPS, HTTP 또는 FTP:** 설치 트리는 HTTP, HTTPS 또는 FTP를 통해 액세스할 수 있는 네트워크 위치에 있습니다.

6.2. 네트워크 기반 설치를 위한 포트

다음 표에는 각 네트워크 기반 설치 유형에 대한 파일을 제공하는 서버에서 열어야 하는 포트가 나열되어 있습니다.

표 6.1. 네트워크 기반 설치를 위한 포트

사용된 프로토콜	오픈할 포트
HTTP	80
HTTPS	443
FTP	21
NFS	2049, 111, 20048
TFTP	69

추가 리소스

- 네트워크 보안

6.3. NFS 서버에서 설치 소스 생성

이 프로세스의 단계에 따라 NFS 서버에 설치 소스를 배치합니다. 물리적 미디어에 연결하지 않고도 이 설치 방법을 사용하여 단일 소스에서 여러 시스템을 설치합니다.

사전 요구 사항

- Red Hat Enterprise Linux 8을 사용하는 서버에 대한 관리자 수준의 액세스 권한이 있으며 이 서버는 설치할 시스템과 동일한 네트워크에 있습니다.
- 바이너리 DVD 이미지를 다운로드했습니다. 자세한 내용은 *표준 RHEL 8 설치 문서에서 설치 ISO 이미지* 다운로드를 참조하십시오.
- 이미지 파일에서 부팅 가능한 CD, DVD 또는 USB 장치를 생성했습니다. 자세한 내용은 *표준 RHEL 8 설치 문서에서 설치* 미디어 생성을 참조하십시오.
- 방화벽을 통해 설치 중인 시스템에서 원격 설치 소스에 액세스할 수 있음을 확인했습니다. 자세한 내용은 *표준 RHEL 8 설치 문서에서 네트워크 기반 설치* 용 포트를 참조하십시오.

절차

1. **nfs-utils** 패키지를 설치합니다.

```
# yum install nfs-utils
```

2. DVD ISO 이미지를 NFS 서버의 디렉터리에 복사합니다.
3. 텍스트 편집기를 사용하여 **/etc/exports** 파일을 열고 다음 구문으로 행을 추가합니다.

```
/exported_directory/ clients
```

4. */exported_directory/*를 디렉토리의 전체 경로로 ISO 이미지로 바꿉니다. *클라이언트를* 대상 시스템의 호스트 이름 또는 IP 주소, 모든 대상 시스템에서 ISO 이미지에 액세스하는 데 사용할 수 있는 서브네트워크 또는 NFS 서버에 대한 네트워크 액세스 권한이 있는 시스템이 ISO 이미지를 사

용할 수 있도록 하려면 별표 기호(*)로 바꿉니다. 이 필드의 형식에 대한 자세한 내용은 **exports(5)** 도움말 페이지를 참조하십시오.
/rhel8-install/ 디렉토리를 모든 클라이언트에 읽기 전용으로 사용할 수 있도록 하는 기본 구성은 다음과 같습니다.

```
/rhel8-install *
```

5. **/etc/exports** 파일을 저장하고 텍스트 편집기를 종료합니다.

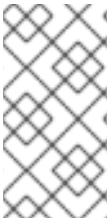
6. nfs 서비스를 시작합니다.

```
# systemctl start nfs-server.service
```

/etc/exports 파일을 변경하기 전에 서비스가 실행 중인 경우 실행 중인 NFS 서버에 대해 다음 명령을 실행하여 구성을 다시 로드합니다.

```
# systemctl reload nfs-server.service
```

이제 NFS를 통해 ISO 이미지에 액세스할 수 있으며 설치 소스로 사용할 준비가 되었습니다.



참고

설치 소스를 구성할 때 **nfs:** 를 프로토콜, 서버 호스트 이름 또는 IP 주소, 콜론 기호 **(:)** 및 ISO 이미지를 포함하는 디렉터리를 사용합니다. 예를 들어 서버 호스트 이름이 **myserver.example.com**이고 ISO 이미지를 **/rhel8-install/**에 저장한 경우 설치 소스로 **nfs:myserver.example.com:/rhel8-install/**을 지정합니다.

6.4. HTTP 또는 HTTPS를 사용하여 설치 소스 생성

다음 절차에 따라 DVD ISO 이미지 및 유효한 **.treeinfo** 파일의 콘텐츠를 포함하는 디렉터리인 설치 트리를 사용하여 네트워크 기반 설치에 대한 설치 소스를 생성합니다. 설치 소스는 HTTP 또는 HTTPS를 통해 액세스할 수 있습니다.

사전 요구 사항

- Red Hat Enterprise Linux 8을 사용하는 서버에 대한 관리자 수준의 액세스 권한이 있으며 이 서버는 설치할 시스템과 동일한 네트워크에 있습니다.
- 바이너리 DVD 이미지를 다운로드했습니다. 자세한 내용은 *표준 RHEL 8 설치 문서에서 설치 ISO 이미지* 다운로드를 참조하십시오.
- 이미지 파일에서 부팅 가능한 CD, DVD 또는 USB 장치를 생성했습니다. 자세한 내용은 *표준 RHEL 8 설치 문서에서 설치* 미디어 생성을 참조하십시오.
- 방화벽을 통해 설치 중인 시스템에서 원격 설치 소스에 액세스할 수 있음을 확인했습니다. 자세한 내용은 *표준 RHEL 8 설치 문서에서 네트워크 기반 설치* 용 포트를 참조하십시오.

절차

1. **http**를 사용하여 설치 소스를 생성하려면 **httpd** 패키지를 설치합니다.

```
# yum install httpd
```

https를 사용하여 설치 소스를 생성하려면 **httpd** 및 **mod_ssl** 패키지를 설치합니다.

```
# yum install httpd mod_ssl
```

주의

Apache 웹 서버 구성이 SSL 보안을 활성화하는 경우 TLSv1 프로토콜만 활성화하고 SSLv2 및 SSLv3을 비활성화합니다. 이는 POODLE SSL 취약점 (CVE-2014-3566) 때문입니다. 자세한 내용은 https://access.redhat.com/solutions/1232413을 참조하십시오.

중요

자체 서명 인증서가 있는 HTTPS 서버를 사용하는 경우 **noverifyssl** 옵션을 사용하여 설치 프로그램을 부팅해야 합니다.

2. DVD ISO 이미지를 HTTP(S) 서버에 복사합니다.

3. **mount** 명령을 사용하여 DVD ISO 이미지를 적절한 디렉터리에 마운트합니다.

```
# mkdir /mnt/rhel8-install/
# mount -o loop,ro -t iso9660 /image_directory/image.iso /mnt/rhel8-install/
```

*/image_directory/image.iso*를 DVD ISO 이미지의 경로로 바꿉니다.

4. 마운트된 이미지의 파일을 HTTP(S) 서버 루트로 복사합니다. 이 명령은 이미지 콘텐츠를 사용하여 **/var/www/html/rhel8-install/** 디렉터리를 생성합니다.

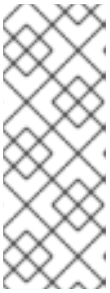
```
# cp -r /mnt/rhel8-install/ /var/www/html/
```

이 명령은 이미지 콘텐츠를 사용하여 **/var/www/html/rhel8-install/** 디렉터리를 생성합니다. 일부 복사 메서드는 유효한 설치 소스에 필요한 **.treeinfo** 파일을 건너뛸 수 있습니다. 이 절차에 표시된 대로 전체 디렉토리에 대해 **cp** 명령을 실행하면 **.treeinfo**가 올바르게 복사됩니다.

5. **httpd** 서비스를 시작합니다.

```
# systemctl start httpd.service
```

이제 설치 트리에 액세스할 수 있으며 설치 소스로 사용할 준비가 되었습니다.



참고

설치 소스를 구성할 때 HTTP 서버 루트를 기준으로 **http://** 또는 **https://** 를 프로토콜, 서버 호스트 이름 또는 IP 주소, ISO 이미지의 파일이 포함된 디렉터리를 사용합니다. 예를 들어 HTTP를 사용하는 경우 서버 호스트 이름은 **myserver.example.com** 이고 이미지에서 **/var/www/html/rhel8-install/** 로 파일을 복사한 경우 설치 소스로 **http://myserver.example.com/rhel8-install/** 를 지정합니다.

추가 리소스

- 다양한 유형의 서버 배포

6.5. FTP를 사용하여 설치 소스 생성

다음 절차에 따라 DVD ISO 이미지 및 유효한 **.treeinfo** 파일의 콘텐츠를 포함하는 디렉터리인 설치 트리를 사용하여 네트워크 기반 설치에 대한 설치 소스를 생성합니다. 설치 소스는 FTP를 통해 액세스할 수 있습니다.

사전 요구 사항

- Red Hat Enterprise Linux 8을 사용하는 서버에 대한 관리자 수준의 액세스 권한이 있으며 이 서버는 설치할 시스템과 동일한 네트워크에 있습니다.
- 바이너리 DVD 이미지를 다운로드했습니다. 자세한 내용은 *표준 RHEL 8 설치 문서에서 설치 ISO 이미지* 다운로드를 참조하십시오.
- 이미지 파일에서 부팅 가능한 CD, DVD 또는 USB 장치를 생성했습니다. 자세한 내용은 *표준 RHEL 8 설치 문서에서 설치* 미디어 생성을 참조하십시오.
- 방화벽을 통해 설치 중인 시스템에서 원격 설치 소스에 액세스할 수 있음을 확인했습니다. 자세한 내용은 *표준 RHEL 8 설치 문서에서 네트워크 기반 설치* 용 포트를 참조하십시오.

절차

1. root로 다음 명령을 실행하여 **vsftpd** 패키지를 설치합니다.

```
# yum install vsftpd
```

2. 텍스트 편집기에서 **/etc/vsftpd/vsftpd.conf** 구성 파일을 열고 편집합니다.
   a. **anonymous_enable=NO** 행을 **anonymous_enable=YES**로 변경합니다.
   b. **write_enable=YES** 행을 **write_enable=NO**로 변경합니다.
   c. **pasv_min_port=*min_port*** 및 **pasv_max_port=*max_port*** 행을 추가합니다. *min_port* 및 *max_port* 를 Passive 모드에서 FTP 서버에서 사용하는 포트 번호 범위(예: **10021** 및 **10031**)로 바꿉니다.
   이 단계는 다양한 방화벽/NAT 설정을 갖춘 네트워크 환경에서 필요할 수 있습니다.
   d. 선택적으로 구성에 사용자 지정 변경 사항을 추가합니다. 사용 가능한 옵션은 **vsftpd.conf(5)** 도움말 페이지를 참조하십시오. 이 절차에서는 기본 옵션이 사용된다고 가정합니다.

주의

vsftpd.conf 파일에 SSL/TLS 보안을 구성한 경우 TLSv1 프로토콜만 활성화하고 SSLv2 및 SSLv3을 비활성화해야 합니다. 이는 POODLE SSL 취약점(CVE-2014-3566) 때문입니다. 자세한 내용은 https://access.redhat.com/solutions/1234773을 참조하십시오.

3. 서버 방화벽을 구성합니다.

   a. 방화벽을 활성화합니다.

   ```
   # systemctl enable firewalld
   # systemctl start firewalld
   ```

   b. 방화벽에서 이전 단계의 FTP 포트 및 포트 범위를 활성화합니다.

   ```
   # firewall-cmd --add-port min_port-max_port/tcp --permanent
   # firewall-cmd --add-service ftp --permanent
   # firewall-cmd --reload
   ```

   *min_port-max_port*를 **/etc/vsftpd/vsftpd.conf** 구성 파일에 입력한 포트 번호로 바꿉니다.

4. DVD ISO 이미지를 FTP 서버에 복사합니다.

5. mount 명령을 사용하여 DVD ISO 이미지를 적절한 디렉터리에 마운트합니다.

   ```
   # mkdir /mnt/rhel8-install
   # mount -o loop,ro -t iso9660 /image-directory/image.iso /mnt/rhel8-install
   ```

   /image-directory/image.iso 를 DVD ISO 이미지의 경로로 바꿉니다.

6. 마운트된 이미지의 파일을 FTP 서버 루트로 복사합니다.

   ```
   # mkdir /var/ftp/rhel8-install
   # cp -r /mnt/rhel8-install/ /var/ftp/
   ```

   이 명령은 이미지 내용이 포함된 **/var/ftp/rhel8-install/** 디렉터리를 생성합니다. 일부 복사 메서드는 유효한 설치 소스에 필요한 **.treeinfo** 파일을 건너뛸 수 있습니다. 이 절차에 표시된 대로 전체 디렉토리에 대해 **cp** 명령을 실행하면 **.treeinfo**가 올바르게 복사됩니다.

7. 복사된 콘텐츠에 올바른 SELinux 컨텍스트 및 액세스 모드가 설정되어 있는지 확인합니다.

   ```
   # restorecon -r /var/ftp/rhel8-install
   # find /var/ftp/rhel8-install -type f -exec chmod 444 {} \;
   # find /var/ftp/rhel8-install -type d -exec chmod 755 {} \;
   ```

8. **vsftpd** 서비스를 시작합니다.

   ```
   # systemctl start vsftpd.service
   ```

   /etc/vsftpd/vsftpd.conf 파일을 변경하기 전에 서비스가 실행 중인 경우 서비스를 다시 시작하여 편집된 파일을 로드합니다.

   ```
   # systemctl restart vsftpd.service
   ```

   부팅 프로세스 중에 시작되도록 **vsftpd** 서비스를 활성화합니다.

   ```
   # systemctl enable vsftpd
   ```

   이제 설치 트리에 액세스할 수 있으며 설치 소스로 사용할 준비가 되었습니다.

참고

설치 소스를 구성할 때 **ftp://**를 프로토콜, 서버 호스트 이름 또는 IP 주소 및 FTP 서버 루트와 관련하여 ISO 이미지의 파일을 저장한 디렉터리를 사용합니다. 예를 들어 서버 호스트 이름이 **myserver.example.com** 이고 이미지에서 **/var/ftp/rhel8-install/** 로 파일을 복사한 경우 설치 소스로 **ftp://myserver.example.com/rhel8-install/** 를 지정합니다.

7장. KICKSTART 설치 시작

여러 가지 방법으로 Kickstart 설치를 시작할 수 있습니다.

- 설치 프로그램 부팅 메뉴를 입력하고 Kickstart 파일을 포함한 옵션을 지정하여 수동으로 수행합니다.
- PXE 부팅에서 부팅 옵션을 편집하여 자동으로 수행합니다.
- 특정 이름으로 볼륨에 파일을 제공하여 자동으로 수행합니다.

다음 섹션에서 이러한 각 방법을 수행하는 방법에 대해 알아봅니다.

7.1. 수동으로 KICKSTART 설치 시작

이 섹션에서는 Kickstart 설치를 수동으로 시작하는 방법을 설명합니다. 즉, 일부 사용자 상호 작용이 필요합니다(**boot:** 프롬프트에서 부팅 옵션 추가). 설치 시스템을 부팅할 때 부팅 옵션 **inst.ks=*location***을 사용하고 위치를 Kickstart 파일의 위치로 교체합니다. 부팅 옵션을 지정하는 정확한 방법과 부팅 프롬프트의 유형은 시스템의 아키텍처에 따라 다릅니다. 자세한 내용은 RHEL 설치 프로그램 부팅 옵션 가이드를 참조하십시오.

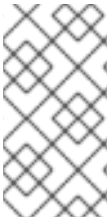
사전 요구 사항

- 설치할 시스템에서 액세스할 수 있는 위치에 Kickstart 파일이 준비되어 있습니다.

절차

1. 로컬 미디어(CD, DVD 또는 USB 플래시 드라이브)를 사용하여 시스템을 부팅합니다.
2. 부팅 프롬프트에서 필요한 부팅 옵션을 지정합니다.
   a. Kickstart 파일 또는 필수 리포지토리가 네트워크 위치에 있는 경우 **ip=** 옵션을 사용하여 네트워크를 구성해야 할 수 있습니다. 설치 프로그램에서 이 옵션 없이 기본적으로 DHCP 프로토콜을 사용하여 모든 네트워크 장치를 구성하려고 합니다.
   b. **inst.ks=** 부팅 옵션과 Kickstart 파일의 위치를 추가합니다.
   c. 필요한 패키지를 설치할 소프트웨어 소스에 액세스하려면 **inst.repo=** 옵션을 추가해야 할 수도 있습니다. 이 옵션을 지정하지 않으면 Kickstart 파일에 설치 소스를 지정해야 합니다.

   부팅 옵션 편집에 대한 자세한 내용은 부팅 옵션 편집을 참조하십시오.
3. 추가된 부팅 옵션을 확인하여 설치를 시작합니다.
   Kickstart 파일에 지정된 옵션을 사용하여 설치가 시작됩니다. Kickstart 파일이 유효하고 모든 필수 명령이 포함된 경우 이 시점에서 설치가 완전히 자동화됩니다.



참고

UEFI Secure Boot가 활성화된 시스템에서 Red Hat Enterprise Linux Beta 릴리스를 설치한 경우 Beta 공개 키를 시스템의 MOK(Machine Owner Key) 목록에 추가합니다. UEFI Secure Boot 및 Red Hat Enterprise Linux 베타 릴리스에 대한 자세한 내용은 UEFI 보안 부팅을 사용하여 베타 시스템 부팅을 참조하십시오.

7.2. PXE를 사용하여 자동으로 KICKSTART 설치 시작

AMD64, Intel 64 및 64비트 ARM 시스템 및 IBM Power Systems 서버는 PXE 서버를 사용하여 부팅할 수 있습니다. PXE 서버를 구성할 때 부트 로더 구성 파일에 부트 옵션을 추가하면 자동으로 설치를 시작할 수 있습니다. 이 방법을 사용하면 부팅 프로세스를 포함하여 설치를 완전히 자동화할 수 있습니다.

이 절차는 일반적인 참조로 제공됩니다. 시스템의 아키텍처에 따라 자세한 단계가 다르며 모든 아키텍처에서 모든 옵션을 사용할 수있는 것은 아닙니다(예: 64비트 IBM Z에서 PXE 부팅을 사용할 수 없음).

사전 요구 사항

- 설치할 위치에 Kickstart 파일이 있어야 합니다.
- 시스템을 부팅하고 설치를 시작하는 데 사용할 수 있는 PXE 서버가 있어야 합니다.

절차

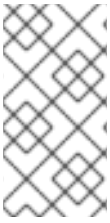
1. PXE 서버에서 부트 로더 구성 파일을 열고 **inst.ks=** 부트 옵션을 적절한 행에 추가합니다. 파일 및 해당 구문의 이름은 시스템의 아키텍처 및 하드웨어에 따라 다릅니다.
   - BIOS가 있는 AMD64 및 Intel 64 시스템에서 파일 이름은 기본 또는 시스템의 IP 주소를 기반으로 할 수 있습니다. 이 경우 설치 항목의 append 행에 **inst.ks=** 옵션을 추가합니다. 구성 파일의 샘플 추가 행은 다음과 유사합니다.

     ```
     append initrd=initrd.img inst.ks=http://10.32.5.1/mnt/archive/RHEL-8/8.x/x86_64/kickstarts/ks.cfg
     ```

   - UEFI 펌웨어 및 IBM Power Systems 서버가 있는 GRUB2 부트 로더(AMD64, Intel 64 및 64비트 ARM 시스템)를 사용하는 시스템에서 파일 이름은 **grub.cfg**가 됩니다. 이 파일에서 설치 항목의 kernel 행에 **inst.ks=** 옵션을 추가합니다. 설정 파일의 샘플 커널 행은 다음과 유사합니다.

     ```
     kernel vmlinuz inst.ks=http://10.32.5.1/mnt/archive/RHEL-8/8.x/x86_64/kickstarts/ks.cfg
     ```

2. 네트워크 서버에서 설치를 부팅합니다.
   이제 Kickstart 파일에 지정된 설치 옵션을 사용하여 설치가 시작됩니다. Kickstart 파일이 유효하고 모든 필수 명령이 포함된 경우 설치가 완전히 자동화됩니다.



참고

UEFI Secure Boot가 활성화된 시스템에서 Red Hat Enterprise Linux Beta 릴리스를 설치한 경우 Beta 공개 키를 시스템의 MOK(Machine Owner Key) 목록에 추가합니다. UEFI Secure Boot 및 Red Hat Enterprise Linux 베타 릴리스에 대한 자세한 내용은 UEFI 보안 부팅을 사용하여 베타 시스템 부팅을 참조하십시오.

추가 리소스

- PXE 서버 설정에 대한 자세한 내용은 netwrok 설치 준비를 참조하십시오.

7.3. 로컬 볼륨을 사용하여 자동으로 KICKSTART 설치 시작

특정 이름이 지정된 스토리지 볼륨에 특정 이름으로 Kickstart 파일을 배치하여 Kickstart 설치를 시작할 수 있습니다.

사전 요구 사항

- 이름이 **OEMDRV** 이고 Kickstart 파일은 **ks.cfg**로 루트에 있는 상태로 준비된 볼륨이 있어야 합니다.
- 이 볼륨을 포함하는 드라이브는 설치 프로그램이 부팅될 때 시스템에서 사용할 수 있어야 합니다.

절차

1. 로컬 미디어(CD, DVD 또는 USB 플래시 드라이브)를 사용하여 시스템을 부팅합니다.
2. 부팅 프롬프트에서 필요한 부팅 옵션을 지정합니다.
   a. 필수 리포지토리가 네트워크 위치에 있는 경우 **ip=** 옵션을 사용하여 네트워크를 구성해야 할 수 있습니다. 설치 프로그램에서 이 옵션 없이 기본적으로 DHCP 프로토콜을 사용하여 모든 네트워크 장치를 구성하려고 합니다.
   b. 필요한 패키지를 설치할 소프트웨어 소스에 액세스하려면 **inst.repo=** 옵션을 추가해야 할 수도 있습니다. 이 옵션을 지정하지 않으면 Kickstart 파일에 설치 소스를 지정해야 합니다.
3. 추가된 부팅 옵션을 확인하여 설치를 시작합니다.
   이제 설치가 시작되고 Kickstart 파일이 자동으로 탐지되어 자동화된 Kickstart 설치를 시작하는 데 사용됩니다.

참고

UEFI Secure Boot가 활성화된 시스템에서 Red Hat Enterprise Linux Beta 릴리스를 설치한 경우 Beta 공개 키를 시스템의 MOK(Machine Owner Key) 목록에 추가합니다. UEFI Secure Boot 및 Red Hat Enterprise Linux 베타 릴리스에 대한 자세한 내용은 UEFI Secure Boot를 사용하여 베타 시스템 부팅을 참조하십시오.

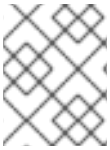
8장. 설치 중에 콘솔 및 로깅

Red Hat Enterprise Linux 설치 프로그램은 **tmux** 터미널 멀티플렉서를 사용하여 기본 인터페이스 외에도 여러 창을 표시하고 제어합니다. 이러한 창은 각각 다른 용도로 사용됩니다. 여러 다른 로그를 표시하며 이는 설치 프로세스 중 문제를 해결하는 데 사용할 수 있습니다. 이 프롬프트가 부팅 옵션 또는 Kickstart 명령을 사용하여 구체적으로 비활성화되지 않은 한 창 중 하나는 **root** 권한이 있는 대화형 쉘 프롬프트를 제공합니다.

참고

일반적으로 설치 문제를 진단하지 않는 한 기본 그래픽 설치 환경을 종료할 이유가 없습니다.

터미널 멀티플렉서는 가상 콘솔 1에서 실행됩니다. 실제 설치 환경에서 **tmux**로 전환하려면 **Ctrl**+**Alt**+**F1**을 누릅니다. 가상 콘솔 6에서 실행되는 기본 설치 인터페이스로 돌아가려면 **Ctrl**+**Alt**+**F6**를 누릅니다.



참고

텍스트 모드 설치를 선택하는 경우 가상 콘솔 1 (**tmux**)에서 시작하며 콘솔 6으로 전환하면 그래픽 인터페이스 대신 쉘 프롬프트가 열립니다.

tmux를 실행하는 콘솔에는 5개의 사용 가능한 창이 있습니다. 해당 내용은 키보드 바로 가기와 함께 다음 표에 설명되어 있습니다. 키보드 바로 가기는 두 부분으로 구성됩니다. 먼저 **Ctrl**+**b** 누른 다음 두 키를 모두 놓고 사용할 창의 숫자 키를 누릅니다.

Ctrl+**b**,**Alt**+ **Tab**, **Ctrl**+**b p** 를 사용하여 각각 다음 또는 이전 **tmux** 창으로 전환할 수도 있습니다.

표 8.1. 사용 가능한 tmux 창

바로 가기	내용
Ctrl+**b 1**	기본 설치 프로그램 창입니다. 텍스트 기반 프롬프트 (텍스트 모드 설치 확인 또는 VNC 직접 모드를 사용하는 경우) 및 일부 디버깅 정보를 포함합니다.
Ctrl+**b 2**	**루트** 권한이 있는 대화형 쉘 프롬프트입니다.
Ctrl+**b 3**	설치 로그; **/tmp/anaconda.log**에 저장된 메시지를 표시합니다.
Ctrl+**b 4**	스토리지 로그; **/tmp/storage.log**에 저장된 스토리지 장치 및 구성과 관련된 메시지를 표시합니다.
Ctrl+**b 5**	프로그램 로그; 설치 프로세스 중에 실행되는 유틸리티의 메시지를 표시하고 **/tmp/program.log**에 저장됩니다.

9장. KICKSTART 파일 유지 관리

Kickstart 파일에서 자동 검사를 실행할 수 있습니다. 일반적으로 새 Kickstart 파일이 유효한지 또는 문제가 있는 Kickstart 파일이 유효한지 확인하려고 합니다.

9.1. KICKSTART 유지 관리 툴 설치

Kickstart 유지 관리 툴을 사용하려면 해당 툴이 포함된 패키지를 설치해야 합니다.

절차

- pykickstart 패키지를 설치합니다.

```
# yum install pykickstart
```

9.2. KICKSTART 파일 확인

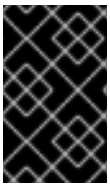
ksvalidator 명령줄 유틸리티를 사용하여 Kickstart 파일이 유효한지 확인합니다. 이 기능은 Kickstart 파일을 광범위하게 변경할 때 유용합니다. **ksvalidator** 명령에서 **-v RHEL8** 옵션을 사용하여 RHEL8 클래스의 새 명령을 승인합니다.

절차

- Kickstart 파일에서 **ksvalidator**를 실행합니다.

```
$ ksvalidator -v RHEL8 /path/to/kickstart.ks
```

*/path/to/kickstart.ks*를 확인할 Kickstart 파일의 경로로 바꿉니다.



중요

검증 툴에서 설치에 성공했는지 보장할 수 없습니다. 이 경우 구문만 올바르며 파일에 더 이상 사용되지 않는 옵션이 포함되지 않습니다. Kickstart 파일의 **%pre,%post** 및 **%packages** 섹션의 유효성을 검증하지 않습니다.

추가 리소스

- *ksvalidator(1)* 도움말 페이지

II 부. CONTENT DELIVERY NETWORK에서 RHEL 등록 및 설치

10장. KICKSTART를 사용하여 CDN에서 RHEL 등록 및 설치

이 섹션에서는 Kickstart를 사용하여 시스템을 등록하고 RHEL 서브스크립션을 연결하며 Red Hat CDN(콘텐츠 전달 네트워크)에서 설치하는 방법에 대해 설명합니다.

10.1. CDN에서 RHEL 등록 및 설치

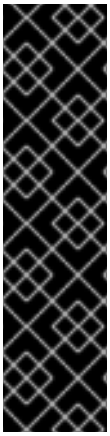
이 절차를 사용하여 시스템을 등록하고 RHEL 서브스크립션을 첨부하고, Red Hat Insights와 Red Hat Insights를 지원하는 **rhsm** Kickstart 명령을 사용하여 Red Hat CDN(Content Delivery Network)에서 설치합니다. **rhsm Kickstart** 명령은 시스템을 등록할 때 사용자 지정 **%post** 스크립트를 사용하는 요구 사항을 제거합니다.

중요

CDN 기능은 부팅 **ISO** 및 **DVD ISO** 이미지 파일에서 지원합니다. 그러나 **Boot ISO** 이미지 파일을 설치 소스로 사용하는 것이 좋습니다. 부팅 **ISO** 이미지 파일의 기본값은 **CDN** 입니다.

사전 요구 사항

- 시스템이 **CDN**에 액세스할 수 있는 네트워크에 연결되어 있습니다.
- **Kickstart** 파일을 생성하고 **HTTP(S), FTP** 또는 **NFS** 서버를 사용하여 이동식 미디어, 하드 드라이브 또는 네트워크 위치에 설치 프로그램에서 사용할 수 있도록 했습니다.
- **Kickstart** 파일은 설치할 시스템에서 액세스할 수 있는 위치에 있습니다.
- 설치를 시작하는 데 사용되는 부팅 미디어를 생성하고 설치 프로그램에서 설치 소스를 사용할 수 있도록 합니다.



중요

- 시스템 등록 후에 사용된 설치 소스 리포지토리는 시스템 부팅 방법에 따라 다릅니다. 자세한 내용은 *표준 RHEL 8 설치 문서의 시스템 등록 후 설치 소스 리포지토리* 섹션을 참조하십시오.
- **Kickstart** 파일에는 서브스크립션이 시스템에서 액세스할 수 있는 **CDN** 하위 집합 및 리포지토리를 제어하므로 리포지토리 구성이 필요하지 않습니다.

절차

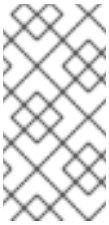
1. **Kickstart** 파일을 엽니다.

2. 파일을 편집하여 파일에 **rhsm Kickstart** 명령과 해당 옵션을 추가합니다.

조직(필수)

조직 **ID**를 입력합니다. 예를 들면 다음과 같습니다.

```
--organization=1234567
```



참고

보안상의 이유로 **CDN**에서 등록 및 설치할 때 **Kickstart**에서 **Red Hat** 사용자 이름 및 암호 계정 세부 정보를 지원하지 않습니다.

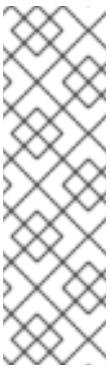
활성키 (필수)

활성화 키를 입력합니다. 활성화 키가 서브스크립션에 등록된 한 여러 개의 키를 입력할 수 있습니다. 예를 들면 다음과 같습니다.

```
--activation-key="Test_key_1" --activation-key="Test_key_2"
```

Red Hat Insights (권장)

대상 시스템을 **Red Hat Insights**에 연결합니다.



참고

Red Hat Insights 는 등록된 **Red Hat** 기반 시스템에 대한 지속적인 심층적인 분석을 제공하여 물리적, 가상 및 클라우드 환경 및 컨테이너 배포 전반에서 보안, 성능 및 안정성에 대한 위협을 사전에 파악할 수 있는 **SaaS**(서비스로서의 소프트웨어)입니다. 설치 프로그램 **GUI**를 사용한 수동 설치와 달리 **Kickstart**를 사용할 때 **Red Hat Insights**에 연결하는 것은 기본적으로 활성화되어 있지 않습니다.

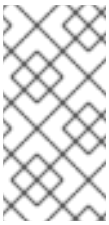
예를 들면 다음과 같습니다.

```
--connect-to-insights
```

HTTP 프록시 (선택 사항)

HTTP 프록시를 설정합니다. 예를 들면 다음과 같습니다.

```
--proxy="user:password@hostname:9000"
```



참고

호스트 이름만 필수입니다. 인증이 없는 기본 포트에서 프록시를 실행해야 하는 경우 옵션은 --proxy="hostname"입니다.

시스템 용도 (선택 사항)

명령을 사용하여 시스템 용도 역할, SLA 및 사용량을 설정합니다.

```
syspurpose --role="Red Hat Enterprise Linux Server" --sla="Premium" --
usage="Production"
```

예제

다음 예제에서는 모든 rhsm Kickstart 명령 옵션이 포함된 최소 Kickstart 파일을 표시합니다.

```
graphical
lang en_US.UTF-8
keyboard us
rootpw 12345
timezone America/New_York
zerombr
clearpart --all --initlabel
autopart
syspurpose --role="Red Hat Enterprise Linux Server" --sla="Premium" --
usage="Production"
rhsm --organization="12345" --activation-key="test_key" --connect-to-insights --
proxy="user:password@hostname:9000"
reboot
%packages
vim
%end
```

3.

Kickstart 파일을 저장하고 설치 프로세스를 시작합니다.

추가 리소스

- 시스템 용도 구성
- Kickstart 설치 시작
- *Red Hat Insights 제품 문서*
- 활성화 키 이해
- HTTP 프록시 사용

10.2. CDN에서 시스템 등록 확인

다음 절차를 사용하여 시스템이 CDN에 등록되어 있는지 확인합니다.

사전 요구 사항

- CDN을 사용하여 등록 및 설치에 설명된 대로 등록 및 설치프로세스를 완료했습니다.
- *Kickstart 설치 시작*에 설명된 대로 Kickstart 설치를 시작했습니다.
- 설치된 시스템이 재부팅되고 터미널 창이 열립니다.

절차

1. 터미널 창에서 root 사용자로 로그인하여 등록을 확인합니다.

```
# subscription-manager list
```

출력에 연결된 서브스크립션 세부 정보가 표시됩니다. 예를 들면 다음과 같습니다.

```
Installed Product Status
```

```
Product Name: Red Hat Enterprise Linux for x86_64
Product ID: 486
Version: X
Arch: x86_64
Status: Subscribed
Status Details
Starts: 11/4/2019
Ends: 11/4/2020
```

2. 자세한 보고서를 보려면 다음 명령을 실행합니다.

```
# subscription-manager list --consumed
```

10.3. CDN에서 시스템 등록 해제

Red Hat CDN에서 시스템을 등록 취소하려면 다음 절차를 사용하십시오.

사전 요구 사항

- CDN에서 RHEL 등록 및 설치에 설명된 대로 등록 및 설치 프로세스를 완료했습니다.
- *Kickstart 설치 시작*에 설명된 대로 Kickstart 설치를 시작했습니다.
- 설치된 시스템이 재부팅되고 터미널 창이 열립니다.

절차

1. 터미널 창에서 root 사용자로 로그인하여 등록 취소합니다.

```
# subscription-manager unregister
```

첨부된 서브스크립션은 시스템에서 등록 취소되고 CDN에 대한 연결이 제거됩니다.

III 부. VNC를 사용하여 원격 RHEL 설치 수행

11장. VNC를 사용하여 원격 RHEL 설치 수행

이 섹션에서는 VNC(Virtual Network Computing)를 사용하여 원격 RHEL 설치를 수행하는 방법에 대해 설명합니다.

11.1. 개요

그래픽 사용자 인터페이스는 PXE를 사용하여 CD, DVD 또는 USB 플래시 드라이브에서 시스템을 부팅하거나 네트워크에서 RHEL을 설치하는 데 권장되는 방법입니다. 그러나 많은 엔터프라이즈 시스템(예: IBM Power Systems 및 64비트 IBM Z)은 자율적으로 실행되며 디스플레이, 키보드 및 마우스에 연결되지 않은 원격 데이터 센터 환경에 있습니다. 이러한 시스템은 종종 *헤드리스 시스템* 이라고하며 일반적으로 네트워크 연결을 통해 제어됩니다. RHEL 설치 프로그램에는 대상 시스템에서 그래픽 설치를 실행하는 VNC(Virtual Network Computing) 설치가 포함되어 있지만 그래픽 설치의 제어는 네트워크의 다른 시스템에서 처리합니다. RHEL 설치 프로그램은 두 가지 VNC 설치 모드인 Direct 및 Connect 를 제공합니다. 연결이 설정되면 두 모드가 다릅니다. 선택한 모드는 환경에 따라 다릅니다.

직접 모드

직접 모드에서 RHEL 설치 프로그램은 대상 시스템에서 시작하도록 구성되어 진행하기 전에 다른 시스템에 설치된 VNC 뷰어를 기다립니다. 직접 모드 설치의 일부로 대상 시스템에 IP 주소와 포트가 표시됩니다. VNC 뷰어를 사용하여 IP 주소와 포트를 사용하여 원격으로 대상 시스템에 연결하고 그래픽 설치를 완료할 수 있습니다.

연결 모드

연결 모드에서 VNC 뷰어는 *수신* 모드의 원격 시스템에서 시작됩니다. VNC 뷰어는 지정된 포트의 대상 시스템에서 들어오는 연결을 기다립니다. RHEL 설치 프로그램이 대상 시스템에서 시작하면 부팅 옵션 또는 Kickstart 명령을 사용하여 시스템 호스트 이름과 포트 번호를 제공합니다. 그런 다음 설치 프로그램은 지정된 시스템 호스트 이름과 포트 번호를 사용하여 수신 대기 VNC 뷰어와 연결을 설정합니다. 연결 모드를 사용하려면 수신 대기 VNC 뷰어가 있는 시스템에서 들어오는 네트워크 연결을 수락할 수 있어야 합니다.

11.2. 고려 사항

VNC를 사용하여 원격 RHEL 설치를 수행할 때 다음 항목을 고려하십시오.

- VNC 클라이언트 애플리케이션: VNC 클라이언트 애플리케이션은 VNC 직접 및 연결 설치를 모두 수행해야 합니다. VNC 클라이언트 애플리케이션은 대부분의 Linux 배포판의 리포지토리에서 사용할 수 있으며 무료 VNC 클라이언트 애플리케이션도 Windows와 같은 다른 운영 체제에서도 사용할 수 있습니다. RHEL에서는 다음 VNC 클라이언트 애플리케이션을 사용할 수 있습니다.
  - 호랑이 vnc는 데스크탑 환경과 독립적이며, playbooksvnc 패키지의 일부로 설치됩니

다.

- vinagre는 GNOME 데스크탑 환경의 일부이며 vinagre 패키지의 일부로 설치됩니다.

참고

VNC 서버는 설치 프로그램에 포함되어 있으므로 설치할 필요가 없습니다.

- 네트워크 및 방화벽:
  - 대상 시스템이 방화벽에서 인바운드 연결을 허용하지 않는 경우 Connect 모드를 사용하거나 방화벽을 비활성화해야 합니다. 방화벽을 비활성화하면 보안에 영향을 미칠 수 있습니다.
  - VNC 뷰어를 실행 중인 시스템이 방화벽에서 들어오는 연결을 허용하지 않는 경우 직접 모드를 사용하거나 방화벽을 비활성화해야 합니다. 방화벽을 비활성화하면 보안에 영향을 미칠 수 있습니다. 방화벽 구성에 대한 자세한 내용은 보안 강화 문서를 참조하십시오.
- 사용자 지정 부팅 옵션: VNC 설치를 시작하려면 사용자 지정 부팅 옵션을 지정해야 하며 시스템 아키텍처에 따라 설치 지침이 다를 수 있습니다.
- Kickstart 설치의 VNC: Kickstart 설치에서 VNC별 명령을 사용할 수 있습니다. vnc 명령만 사용하면 직접 모드에서 RHEL 설치가 실행됩니다. 연결 모드를 사용하여 설치를 설정하는 데 추가 옵션을 사용할 수 있습니다.

11.3. VNC 직접 모드에서 원격 RHEL 설치 수행

VNC 직접 모드에서 원격 RHEL 설치를 수행하려면 다음 절차를 사용하십시오. 직접 모드에서는 VNC 뷰어가 RHEL;과 함께 설치 중인 대상 시스템에 대한 연결을 시작할 것으로 예상합니다. 이 절차에서는 VNC 뷰어를 사용하는 시스템을 원격 시스템이라고 합니다. RHEL 설치 프로그램에서 원격 시스템의 VNC 뷰어에서 대상 시스템으로의 연결을 시작하라는 메시지가 표시됩니다.

참고

이 절차에서는 VNC 뷰어로 TigerVNC를 사용합니다. 다른 뷰어에 대한 특정 지침은 다를 수 있지만 일반적인 원칙이 적용됩니다.

사전 요구 사항

- 예를 들어 root로 원격 시스템에 VNC 뷰어가 설치되어 있습니다.

```
# yum install tigervnc
```

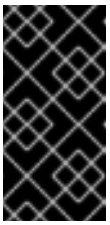
- 네트워크 부팅 서버를 설정하고 대상 시스템에서 설치를 부팅했습니다.

절차

1. 대상 시스템의 RHEL 부팅 메뉴에서 키보드의 Tab 키를 눌러 부팅 옵션을 편집합니다.

2. inst.vnc 옵션을 명령줄 끝에 추가합니다.

   a. 설치 중인 시스템에 대한 VNC 액세스를 제한하려면 명령행 끝에 inst.vncpassword=PASSWORD 부팅 옵션을 추가합니다. 설치에 사용할 암호로 PASSWORD를 바꿉니다. VNC 암호는 6에서 8자 사이여야 합니다.



중요

inst.vncpassword= 옵션에 임시 암호를 사용합니다. 기존 또는 root 암호가 아니어야 합니다.

3. Enter를 눌러 설치를 시작합니다. 대상 시스템은 설치 프로그램을 초기화하고 필요한 서비스를 시작합니다. 시스템이 준비되면 시스템의 IP 주소 및 포트 번호를 제공하는 메시지가 표시됩니다.

4. 원격 시스템에서 VNC 뷰어를 엽니다.

5. VNC 서버 필드에 IP 주소와 포트 번호를 입력합니다.

6. 연결을 클릭합니다.

7. VNC 암호를 입력하고 OK를 클릭합니다. VNC 연결이 설정된 새 창이 열리고 RHEL 설치 메

뉴를 표시합니다. 이 창에서 그래픽 사용자 인터페이스를 사용하여 대상 시스템에 **RHEL**을 설치할 수 있습니다.

11.4. VNC 연결 모드에서 원격 RHEL 설치 수행

VNC Connect 모드에서 원격 **RHEL** 설치를 수행하려면 다음 절차를 사용하십시오. 연결 모드에서 **RHEL**을 사용하여 설치 중인 대상 시스템은 다른 시스템에 설치된 **VNC** 뷰어에 대한 연결을 시작합니다. 이 절차에서는 **VNC** 뷰어를 사용하는 시스템을 원격 시스템이라고 합니다.

참고

이 절차에서는 **VNC** 뷰어로 **TigerVNC**를 사용합니다. 다른 뷰어에 대한 특정 지침은 다를 수 있지만 일반적인 원칙이 적용됩니다.

사전 요구 사항

- 예를 들어 **root**로 원격 시스템에 **VNC** 뷰어가 설치되어 있습니다.

```
# yum install tigervnc
```

- 대상 시스템에서 설치를 시작하도록 네트워크 부팅 서버를 설정했습니다.
- **VNC Connect** 설치에 부팅 옵션을 사용하도록 대상 시스템을 구성했습니다.
- **VNC** 뷰어가 있는 원격 시스템이 필수 포트에서 들어오는 연결을 수락하도록 구성되어 있는지 확인했습니다. 확인은 네트워크 및 시스템 구성에 따라 다릅니다. 자세한 내용은 보안 강화 및 보안 네트워크 문서를 참조하십시오.

절차

1. 다음 명령을 실행하여 *청취 모드에서* 원격 시스템에서 **VNC** 뷰어를 시작합니다.

```
$ vncviewer -listen PORT
```

2. **PORT**를 연결에 사용된 포트 번호로 바꿉니다.

3. 터미널에는 대상 시스템에서 들어오는 연결을 대기 중임을 나타내는 메시지가 표시됩니다.

```
TigerVNC Viewer 64-bit v1.8.0
Built on: 2017-10-12 09:20
Copyright (C) 1999-2017 TigerVNC Team and many others (see README.txt)
See http://www.tigervnc.org for information on TigerVNC.

Thu Jun 27 11:30:57 2019
 main:       Listening on port 5500
```

4. 네트워크에서 대상 시스템을 부팅합니다.

5. 대상 시스템의 **RHEL** 부팅 메뉴에서 키보드의 **Tab** 키를 눌러 부팅 옵션을 편집합니다.

6. **inst.vnc inst.vncconnect=HOST:PORT** 옵션을 명령줄 끝에 추가합니다.

7. ***HOST*** 를 수신 대기 **VNC** 뷰어를 실행 중인 원격 시스템의 **IP** 주소로 교체하고 **VNC** 뷰어가 수신 대기 중인 포트 번호로 ***PORT*** 를 변경합니다.

8. **Enter**를 눌러 설치를 시작합니다. 시스템이 설치 프로그램을 초기화하고 필요한 서비스를 시작합니다. 초기화 프로세스가 완료되면 설치 프로그램에서 제공된 **IP** 주소 및 포트에 연결을 시도합니다.

9. 연결에 성공하면 **VNC** 연결이 설정된 새 창이 열리고 **RHEL** 설치 메뉴를 표시합니다. 이 창에서 그래픽 사용자 인터페이스를 사용하여 대상 시스템에 **RHEL**을 설치할 수 있습니다.

IV 부. 고급 구성 옵션

12장. 시스템 용도 구성

시스템 용도를 사용하여 **Red Hat Enterprise Linux 8** 시스템의 용도를 기록합니다. 시스템 용도를 설정하면 인타이틀먼트 서버가 가장 적합한 서브스크립션을 자동으로 첨부할 수 있습니다. 이 섹션에서는 **Kickstart**를 사용하여 시스템 용도를 구성하는 방법에 대해 설명합니다.

이점은 다음과 같습니다.

- 시스템 관리자 및 비즈니스 운영에 대한 심층적인 시스템 수준 정보입니다.
- 시스템 생성 이유 및 의도된 용도를 결정할 때 오버헤드 감소.
- 서브스크립션 관리자 자동 연결 및 시스템 사용량에 대한 자동 검색 및 조정에 대한 고객 경험 개선.

12.1. 개요

다음 방법 중 하나로 시스템 용도 데이터를 입력할 수 있습니다.

- 이미지 생성 중
- **Red Hat** 에 연결 화면을 사용하여 시스템을 등록하고 **Red Hat** 서브스크립션을 첨부하는 경우 **GUI** 설치 중
- **syspurpose Kickstart** 명령을 사용할 때 **Kickstart** 설치 중
- **syspurpose CLI(**명령줄**)** 툴을 사용한 설치 후

시스템의 의도된 용도를 기록하기 위해 다음과 같은 시스템 용도 구성 요소를 구성할 수 있습니다. 선택한 값은 시스템에 가장 적합한 서브스크립션을 연결하기 위해 등록 시 인타이틀먼트 서버에서 사용합니다.

Role

- **Red Hat Enterprise Linux Server**
- **Red Hat Enterprise Linux Workstation**
- **Red Hat Enterprise Linux Compute Node**

서비스 수준 계약

- **Premium**
- **Standard**
- **Self-Support**

사용법

- 프로덕션
- 개발/테스트
- 재해 복구

추가 리소스

- 사용자 지정 **RHEL** 시스템 이미지 구성
- 고급 **RHEL 8** 설치 수행
- **Red Hat** 서브스크립션 관리자 사용 및 구성

12.2. KICKSTART 파일에서 시스템 용도 구성

설치 중에 시스템 용도를 구성하려면 다음 절차의 단계를 따르십시오. 이를 위해 **Kickstart** 구성 파일에서 **syspurpose Kickstart** 명령을 사용합니다.

시스템 용도는 **Red Hat Enterprise Linux** 설치 프로그램의 선택적 기능이지만 가장 적합한 서브스크립션을 자동으로 첨부하도록 시스템 용도를 구성하는 것이 좋습니다.

참고

설치가 완료된 후 시스템 용도를 활성화할 수도 있습니다. 이 작업을 수행하려면 **syspurpose** 명령줄 도구를 사용합니다. **syspurpose tool** 명령은 **syspurpose Kickstart** 명령과 다릅니다.

syspurpose Kickstart 명령에 다음 작업을 사용할 수 있습니다.

role

시스템의 의도한 역할을 설정합니다. 이 작업은 다음 형식을 사용합니다.

```
syspurpose --role=
```

할당된 역할은 다음과 같습니다.

- **Red Hat Enterprise Linux Server**
- **Red Hat Enterprise Linux Workstation**
- **Red Hat Enterprise Linux Compute Node**

SLA

시스템의 **SLA**를 설정합니다. 이 작업은 다음 형식을 사용합니다.

```
syspurpose --sla=
```

할당된 Sla는 다음을 수행할 수 있습니다.

- **Premium**
- **Standard**
- **Self-Support**

사용법

시스템의 의도된 사용량을 설정합니다. 이 작업은 다음 형식을 사용합니다.

```
syspurpose --usage=
```

할당된 사용은 다음과 같습니다.

- **Production**
- **Development/Test**
- **Disaster Recovery**

애드온

추가 계층화된 제품 또는 기능입니다. 여러 항목을 추가하려면 계층화된 제품/기능별로 **--addon**을 여러 번 지정합니다. 이 작업은 다음 형식을 사용합니다.

```
syspurpose --addon=
```

12.3. 추가 리소스

- **subscription-manager** 명령줄 툴을 사용하여 시스템 용도 구성

13장. 설치 중 드라이버 업데이트

이 섹션에서는 **Red Hat Enterprise Linux** 설치 프로세스 중에 드라이버 업데이트를 완료하는 방법에 대해 설명합니다.

참고

이는 설치 프로세스의 선택적 단계입니다. 필요한 경우가 아니면 드라이버 업데이트를 수행하지 않는 것이 좋습니다.

13.1. 사전 요구 사항

Red Hat, 하드웨어 벤더 또는 신뢰할 수 있는 타사 벤더에 의해 **Red Hat Enterprise Linux** 설치 중에 드라이버 업데이트가 필요하다는 알림을 받았습니다.

13.2. 개요

Red Hat Enterprise Linux는 많은 하드웨어 장치에 대한 드라이버를 지원하지만 일부 새로 릴리스된 드라이버는 지원되지 않을 수 있습니다. 드라이버 업데이트는 지원되지 않는 드라이버가 설치가 완료되지 않는 경우에만 수행해야 합니다. 일반적으로 설치 중에 드라이버를 업데이트하는 것은 특정 구성을 지원하는 데만 필요합니다. 예를 들어 시스템의 스토리지 장치에 대한 액세스를 제공하는 스토리지 어댑터 카드용 드라이버를 설치합니다.

주의

드라이버 업데이트 디스크에서 충돌하는 커널 드라이버를 비활성화할 수 있습니다. 드문 경우지만 커널 모듈을 언로드하면 설치 오류가 발생할 수 있습니다.

13.3. 드라이버 업데이트 유형

Red Hat, 하드웨어 벤더 또는 신뢰할 수 있는 타사는 드라이버 업데이트를 **ISO** 이미지 파일로 제공합니다. **ISO** 이미지 파일이 표시되면 드라이버 업데이트 유형을 선택합니다.

드라이버 업데이트 유형

자동

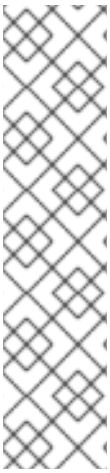
권장 드라이버 업데이트 방법; **OEMDRV** 레이블이 지정된 스토리지 장치(**CD, DVD** 또는 **USB** 플래시 드라이브 포함)는 물리적으로 시스템에 연결됩니다. 설치가 시작될 때 **OEMDRV** 스토리지 장치가 있는 경우 드라이버 업데이트 디스크로 처리되고 설치 프로그램이 해당 드라이버를 자동으로 로드합니다.

지원됨

설치 프로그램에서 드라이버 업데이트를 찾도록 요청합니다. **OEMDRV** 이외의 레이블과 함께 로컬 스토리지 장치를 사용할 수 있습니다. 설치를 시작할 때 **inst.dd** 부팅 옵션이 지정됩니다. 매개 변수 없이 이 옵션을 사용하는 경우 설치 프로그램은 시스템에 연결된 모든 스토리지 장치를 표시하고 드라이버 업데이트가 포함된 장치를 선택하라는 메시지를 표시합니다.

수동

드라이버 업데이트 이미지 또는 **RPM** 패키지의 경로를 수동으로 지정합니다. **OEMDRV** 이외의 레이블과 함께 로컬 스토리지 장치 또는 설치 시스템에서 액세스할 수 있는 네트워크 위치를 사용할 수 있습니다. **inst.dd=location** 부팅 옵션은 설치를 시작할 때 지정됩니다. 여기서 ***location*** 은 드라이버 업데이트 디스크 또는 **ISO** 이미지의 경로입니다. 이 옵션을 지정하면 설치 프로그램에서 지정된 위치에 있는 드라이버 업데이트를 로드하려고 합니다. 수동 드라이버 업데이트를 사용하면 로컬 스토리지 장치 또는 네트워크 위치(**HTTP, HTTPS** 또는 **FTP** 서버)를 지정할 수 있습니다.



참고

- **inst.dd=location** 및 **inst.dd** 둘 다 동시에 사용할 수 있습니다. 여기서 ***location***은 드라이버 업데이트 디스크 또는 **ISO** 이미지의 경로입니다. 이 시나리오에서 설치 프로그램은 위치에서 사용 가능한 드라이버 업데이트를 로드하고 드라이버 업데이트가 포함된 장치를 선택하라는 메시지를 표시합니다.
- 네트워크 위치에서 드라이버 업데이트를 로드할 때 **ip= option**을 사용하여 네트워크를 초기화합니다.

제한 사항

Secure Boot 기술이 활성화된 **UEFI** 시스템에서 모든 드라이버에 유효한 인증서로 서명해야 합니다. **Red Hat** 드라이버는 **Red Hat**의 개인 키 중 하나에서 서명하고 커널의 해당 공개 키로 인증합니다. 추가 드라이버를 로드하는 경우 해당 드라이버가 서명되었는지 확인합니다.

13.4. 드라이버 업데이트 준비

이 절차에서는 **CD** 및 **DVD**에서 드라이버 업데이트를 준비하는 방법을 설명합니다.

사전 요구 사항

- Red Hat, 하드웨어 벤더 또는 신뢰할 수 있는 타사 벤더의 드라이버 업데이트 ISO 이미지를 받았습니다.

- 드라이버 업데이트 ISO 이미지를 CD 또는 DVD로 구웠습니다.

주의

.iso로 끝나는 단일 ISO 이미지 파일만 CD 또는 DVD에서 사용할 수 있는 경우, 굽기 프로세스가 성공하지 못했습니다. ISO 이미지를 CD 또는 DVD에 구울 수 있는 방법에 대한 지침은 시스템 화상을 참조하십시오.

절차

1. 드라이버 업데이트 CD 또는 DVD를 시스템의 CD/DVD 드라이브에 삽입한 후 시스템의 파일 관리자 도구를 사용하여 찾습니다.

2. rhdd3 파일을 사용할 수 있는지 확인합니다. rhdd3 은 드라이버 설명 및 rpms 라는 디렉토리가 포함된 서명 파일로, 다양한 아키텍처의 실제 드라이버가 있는 RPM 패키지를 포함합니다.

13.5. 자동 드라이버 업데이트 수행

다음 절차에서는 설치 중에 자동 드라이버 업데이트를 수행하는 방법을 설명합니다.

사전 요구 사항

- OEMDRV 라벨을 사용하여 표준 디스크 파티션에 드라이버 업데이트 이미지를 배치하거나 OEMDRV 드라이버를 CD 또는 DVD로 업데이트했습니다. 드라이버 업데이트 프로세스 중에 RAID 또는 LVM 볼륨과 같은 고급 스토리지에 액세스할 수 없습니다.

- OEMDRV 볼륨 레이블과 블록 장치를 시스템에 연결하거나 설치 프로세스를 시작하기 전에 준비된 CD 또는 DVD를 시스템의 CD/DVD 드라이브에 삽입했습니다.

절차

1. 전제 조건 단계를 완료하면 설치 프로그램이 시작될 때 드라이버가 자동으로 로드되고 설치 프로세스 중에 시스템에 설치됩니다.

13.6. 지원되는 드라이버 업데이트 수행

다음 절차에서는 설치 중에 지원 드라이버 업데이트를 수행하는 방법을 설명합니다.

사전 요구 사항

OEMDRV 볼륨 레이블이 없는 블록 장치를 시스템에 연결하고 드라이버 디스크 이미지를 이 장치에 복사하거나 드라이버 업데이트 **CD** 또는 **DVD**를 준비하여 설치 프로세스를 시작하기 전에 시스템의 **CD/DVD** 드라이브에 삽입했습니다.

참고

ISO 이미지 파일을 **CD** 또는 **DVD**에 구울 수는 있지만 **OEMDRV** 볼륨 레이블이 없는 경우 인수 없이 **inst.dd** 옵션을 사용할 수 있습니다. 설치 프로그램은 **CD** 또는 **DVD**에서 드라이버를 스캔하고 선택할 수 있는 옵션을 제공합니다. 이 시나리오에서는 설치 프로그램에서 드라이버 업데이트 **ISO** 이미지를 선택하라는 메시지가 표시되지 않습니다. 또 다른 시나리오는 **inst.dd=location** 부팅 옵션과 함께 **CD** 또는 **DVD**를 사용하는 것입니다. 이렇게 하면 설치 프로그램에서 **CD** 또는 **DVD**에서 드라이버 업데이트를 자동으로 스캔할 수 있습니다. 자세한 내용은 수동 드라이버 업데이트 수행을 참조하십시오.

절차

1. 부팅 메뉴 창의 키보드에서 **Tab** 키를 눌러 부팅 명령줄을 표시합니다.

2. **inst.dd** 부팅 옵션을 명령줄에 추가하고 **Enter** 키를 눌러 부팅 프로세스를 실행합니다.

3. 메뉴에서 로컬 디스크 파티션 또는 **CD** 또는 **DVD** 장치를 선택합니다. 설치 프로그램은 **ISO** 파일 또는 드라이버 업데이트 **RPM** 패키지를 스캔합니다.

4. 선택 사항: 드라이버 업데이트 **ISO** 파일을 선택합니다.

참고

선택한 장치 또는 파티션에 ISO 이미지 파일이 아닌 드라이버 업데이트 RPM 패키지가 포함된 경우 이 단계가 필요하지 않습니다(예: 드라이버 업데이트 CD 또는 DVD가 포함된 광기 드라이브).

5. 필요한 드라이버를 선택합니다.

   a. 키보드의 숫자 키를 사용하여 드라이버 선택을 전환합니다.

   b. c를 눌러 선택한 드라이버를 설치합니다. 선택한 드라이버가 로드되고 설치 프로세스가 시작됩니다.

13.7. 수동 드라이버 업데이트 수행

다음 절차에서는 설치 중에 수동 드라이버 업데이트를 수행하는 방법을 설명합니다.

사전 요구 사항

- 드라이버 업데이트 ISO 이미지 파일을 USB 플래시 드라이브 또는 웹 서버에 배치하고 컴퓨터에 연결합니다.

절차

1. 부팅 메뉴 창의 키보드에서 Tab 키를 눌러 부팅 명령줄을 표시합니다.

2. inst.dd=location 부트 옵션을 명령줄에 추가합니다. 여기서 location은 드라이버 업데이트의 경로입니다. 일반적으로 이미지 파일은 웹 서버(예: http://server.example.com/dd.iso) 또는 USB 플래시 드라이브(예: /dev/sdb1)에 있습니다. 드라이버 업데이트가 포함된 RPM 패키지를 지정할 수도 있습니다(예: http://server.example.com/dd.rpm).

3. Enter를 눌러 부팅 프로세스를 실행합니다. 지정된 위치에서 사용할 수 있는 드라이버가 자동으로 로드되고 설치 프로세스가 시작됩니다.

추가 리소스

- inst.dd 부팅 옵션

13.8. 드라이버 비활성화

다음 절차에서는 오작동 드라이버를 비활성화하는 방법을 설명합니다.

사전 요구 사항

- 설치 프로그램 부팅 메뉴를 부팅했습니다.

절차

1. 부팅 메뉴에서 키보드의 **Tab** 키를 눌러 부팅 명령줄을 표시합니다.
2. **modprobe.blacklist=driver_name** 부팅 옵션을 명령줄에 추가합니다.
3. ***driver_name***을 비활성화하려는 드라이버 또는 드라이버 이름으로 교체합니다. 예를 들면 다음과 같습니다.

```
modprobe.blacklist=ahci
```

modprobe.blacklist= boot 옵션을 사용하여 비활성화된 드라이버는 설치된 시스템에서 비활성화된 상태로 남아 있으며 **/etc/modprobe.d/anaconda-blacklist.conf** 파일에 나타납니다.

4. **Enter**를 눌러 부팅 프로세스를 실행합니다.

14장. PXE를 사용하여 네트워크에서 설치 준비

이 섹션에서는 PXE 부팅 및 네트워크 설치를 활성화하도록 PXE 서버에서 TFTP 및 DHCP를 구성하는 방법에 대해 설명합니다.

14.1. 네트워크 설치 개요

네트워크 설치를 통해 설치 서버에 액세스할 수 있는 시스템에 Red Hat Enterprise Linux를 설치할 수 있습니다. 최소한 네트워크 설치에는 두 개의 시스템이 필요합니다.

PXE Server

DHCP 서버, TFTP 서버 및 HTTP, HTTPS, FTP 또는 NFS 서버를 실행하는 시스템. 각 서버는 다른 물리적 시스템에서 실행할 수 있지만 이 섹션의 절차는 단일 시스템이 모든 서버를 실행하고 있다고 가정합니다.

클라이언트

Red Hat Enterprise Linux를 설치하는 시스템. 설치가 시작되면 클라이언트는 DHCP 서버에 쿼리하고, TFTP 서버에서 부팅 파일을 수신하고, HTTP, HTTPS, FTP 또는 NFS 서버에서 설치 이미지를 다운로드합니다. 다른 설치 방법과 달리 클라이언트에는 설치를 시작하는 데 물리적 부팅 미디어가 필요하지 않습니다.

참고

네트워크에서 클라이언트를 부팅하려면 BIOS/UEFI 또는 빠른 부팅 메뉴에서 클라이언트를 구성합니다. 일부 하드웨어에서는 네트워크에서 부팅하는 옵션이 비활성화되거나 사용할 수 없는 경우가 있습니다.

PXE를 사용하여 네트워크에서 Red Hat Enterprise Linux 설치를 준비하는 워크플로우 단계는 다음과 같습니다.

단계

1. 설치 ISO 이미지 또는 설치 트리를 NFS, HTTPS, HTTP 또는 FTP 서버로 내보냅니다.
2. TFTP 서버와 DHCP 서버를 구성하고 PXE 서버에서 TFTP 서비스를 시작합니다.

3. 클라이언트를 부팅하고 설치를 시작합니다.

중요

GRUB2 부트 로더는 TFTP 서버 외에도 HTTP에서 네트워크 부팅을 지원합니다. 이 프로토콜을 통해 커널 및 초기 RAM 디스크 vmlinuz 및 initrd인 부팅 파일을 전송하는 속도가 느려 시간 초과 오류가 발생할 수 있습니다. HTTP 서버에는 이러한 위험이 발생하지 않지만 부트 파일을 보낼 때 TFTP 서버를 사용하는 것이 좋습니다.

추가 리소스

- Kickstart 설치를 위한 설치 소스 생성
- BIOS 기반 클라이언트용 TFTP 서버 구성
- UEFI 기반 클라이언트용 TFTP 서버 구성
- IBM Power 시스템용 네트워크 서버 구성
- Red Hat Satellite 제품 문서

14.2. BIOS 기반 클라이언트용 TFTP 서버 구성

다음 절차를 사용하여 TFTP 서버와 DHCP 서버를 구성하고 BIOS 기반 AMD 및 Intel 64비트 시스템용 PXE 서버에서 TFTP 서비스를 시작합니다.

중요

이 섹션의 모든 구성 파일은 예시입니다. 구성 세부 정보는 아키텍처 및 특정 요구 사항에 따라 다릅니다.

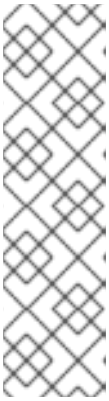
절차

1. root로 다음 패키지를 설치합니다. 네트워크에 DHCP 서버가 이미 구성되어 있는 경우 dhcp-server 패키지를 제외합니다.

```
# yum install tftp-server dhcp-server
```

2. 방화벽에서 **tftp service**에 대한 수신 연결을 허용합니다.

```
# firewall-cmd --add-service=tftp
```



참고

- 이 명령은 다음 서버가 재부팅될 때까지 임시 액세스를 활성화합니다. 영구 액세스를 활성화하려면 명령에 **--permanent** 옵션을 추가합니다.
- 설치 **ISO** 파일의 위치에 따라 **HTTP** 또는 기타 서비스에 대해 들어오는 연결을 허용해야 할 수 있습니다.

3. 다음 예제 **/etc/dhcp/dhcpd.conf** 파일에 표시된 대로 **SYSLINUX**와 함께 패키지된 부팅 이미지를 사용하도록 **DHCP** 서버를 구성합니다. **DHCP** 서버가 이미 구성된 경우 **DHCP** 서버에서 이 단계를 수행합니다.

```
option space pxelinux;
option pxelinux.magic code 208 = string;
option pxelinux.configfile code 209 = text;
option pxelinux.pathprefix code 210 = text;
option pxelinux.reboottime code 211 = unsigned integer 32;
option architecture-type code 93 = unsigned integer 16;

subnet 10.0.0.0 netmask 255.255.255.0 {
 option routers 10.0.0.254;
 range 10.0.0.2 10.0.0.253;

 class "pxeclients" {
   match if substring (option vendor-class-identifier, 0, 9) = "PXEClient";
   next-server 10.0.0.1;

   if option architecture-type = 00:07 {
     filename "BOOTX64.efi";
     } else {
     filename "pxelinux/pxelinux.0";
   }
 }
}
```

4. **DVD ISO** 이미지 파일의 **SYSLINUX** 패키지에서 **pxelinux.0** 파일에 액세스합니다. 여기서 ***my_local_directory***는 생성한 디렉터리의 이름입니다.

```
# mount -t iso9660 /path_to_image/name_of_image.iso /mount_point -o loop,ro
```

```
# cp -pr /mount_point/BaseOS/Packages/syslinux-tftpboot-version-architecture.rpm
/my_local_directory
```

```
# umount /mount_point
```

5. **패키지를 추출합니다.**

```
# rpm2cpio syslinux-tftpboot-version-architecture.rpm | cpio -dimv
```

6. **tftpboot/에 pxelinux/ 디렉터리를 만들고 디렉터리의 모든 파일을 pxelinux/ 디렉터리에 복사합니다.**

```
# mkdir /var/lib/tftpboot/pxelinux
```

```
# cp my_local_directory/tftpboot/* /var/lib/tftpboot/pxelinux
```

7. **pxelinux/ 디렉터리에 pxelinux.cfg/ 디렉터리를 만듭니다.**

```
# mkdir /var/lib/tftpboot/pxelinux/pxelinux.cfg
```

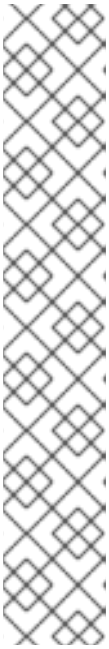
8. **default라는 구성 파일을 생성하고 다음 예와 같이 pxelinux.cfg/ 디렉터리에 추가합니다.**

```
default vesamenu.c32
prompt 1
timeout 600

display boot.msg

label linux
  menu label ^Install system
  menu default
  kernel images/RHEL-8/vmlinuz
  append initrd=images/RHEL-8/initrd.img ip=dhcp inst.repo=http://10.32.5.1/RHEL-
8/x86_64/iso-contents-root/
label vesa
  menu label Install system with ^basic video driver
  kernel images/RHEL-8/vmlinuz
  append initrd=images/RHEL-8/initrd.img ip=dhcp inst.xdriver=vesa nomodeset
inst.repo=http://10.32.5.1/RHEL-8/x86_64/iso-contents-root/
label rescue
  menu label ^Rescue installed system
  kernel images/RHEL-8/vmlinuz
  append initrd=images/RHEL-8/initrd.img rescue
```

```
label local
  menu label Boot from ^local drive
  localboot 0xffff
```



참고

- 설치 프로그램은 런타임 이미지 없이는 부팅할 수 없습니다. **inst.stage2** 부팅 옵션을 사용하여 이미지 위치를 지정합니다. 또는 **inst.repo=** 옵션을 사용하여 이미지와 설치 소스를 지정할 수 있습니다.
- **inst.repo**와 함께 사용되는 설치 소스 위치에는 유효한 **.treeinfo** 파일이 포함되어야 합니다.
- **RHEL8** 설치 **DVD**를 설치 소스로 선택하면 **.treeinfo** 파일은 **BaseOS** 및 **AppStream** 리포지토리를 가리킵니다. 단일 **inst.repo** 옵션을 사용하여 두 리포지토리를 로드할 수 있습니다.

9. **/var/lib/tftpboot/** 디렉터리에 부팅 이미지 파일을 저장하고 부팅 이미지 파일을 디렉터리에 복사합니다. 이 예에서 디렉터리는 **/var/lib/tftpboot/pxelinux/images/RHEL-8/**:입니다.

```
# mkdir -p /var/lib/tftpboot/pxelinux/images/RHEL-8/
# cp /path_to_x86_64_images/pxeboot/{vmlinuz,initrd.img}
/var/lib/tftpboot/pxelinux/images/RHEL-8/
```

10. **DHCP** 서버에서 **dhcpd** 서비스를 시작하고 활성화합니다. **localhost**에 **DHCP** 서버를 구성한 경우 **localhost**에서 **dhcpd** 서비스를 시작하고 활성화합니다.

```
# systemctl start dhcpd
# systemctl enable dhcpd
```

11. **tftp.socket** 서비스를 시작하고 활성화합니다.

```
# systemctl start tftp.socket
# systemctl enable tftp.socket
```

이제 **PXE** 부팅 서버가 **PXE** 클라이언트를 제공할 준비가 되었습니다. **Red Hat Enterprise Linux**를 설치하는 시스템인 클라이언트를 시작하고, 부팅 소스를 지정하라는 메시지가 표시되면 **PXE** 부팅 을 선택한 다음 네트워크 설치를 시작할 수 있습니다.

14.3. UEFI 기반 클라이언트용 TFTP 서버 구성

다음 절차를 사용하여 **TFTP** 서버와 **DHCP** 서버를 구성하고 **UEFI** 기반 **AMD64, Intel 64** 및 **64**비트 **ARM** 시스템의 **PXE** 서버에서 **TFTP** 서비스를 시작합니다.

중요

- 이 섹션의 모든 구성 파일은 예시입니다. 구성 세부 정보는 아키텍처 및 특정 요구 사항에 따라 다릅니다.
- **Red Hat Enterprise Linux 8 UEFI PXE** 부팅은 **MAC** 기반 **grub** 메뉴 파일의 소문자 파일 형식을 지원합니다. 예를 들어 **grub2**의 **MAC** 주소 파일 형식은 **grub.cfg-01-aa-bb-cc-dd-ee-ff**입니다.

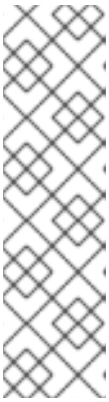
절차

1. **root**로 다음 패키지를 설치합니다. 네트워크에 **DHCP** 서버가 이미 구성되어 있는 경우 **dhcp-server** 패키지를 제외합니다.

```
# yum install tftp-server dhcp-server
```

2. 방화벽에서 **tftp service**에 대한 수신 연결을 허용합니다.

```
# firewall-cmd --add-service=tftp
```



참고

- 이 명령은 다음 서버가 재부팅될 때까지 임시 액세스를 활성화합니다. 영구 액세스를 활성화하려면 명령에 **--permanent** 옵션을 추가합니다.
- 설치 **ISO** 파일의 위치에 따라 **HTTP** 또는 기타 서비스에 대해 들어오는 연결을 허용해야 할 수 있습니다.

3. 다음 예제 **/etc/dhcp/dhcpd.conf** 파일에 표시된 대로 **shim** 과 함께 패키지된 부팅 이미지를 사용하도록 **DHCP** 서버를 구성합니다. **DHCP** 서버가 이미 구성된 경우 **DHCP** 서버에서 이 단계를 수행합니다.

```
option space pxelinux;
option pxelinux.magic code 208 = string;
option pxelinux.configfile code 209 = text;
option pxelinux.pathprefix code 210 = text;
option pxelinux.reboottime code 211 = unsigned integer 32;
option architecture-type code 93 = unsigned integer 16;

subnet 10.0.0.0 netmask 255.255.255.0 {
 option routers 10.0.0.254;
 range 10.0.0.2 10.0.0.253;

 class "pxeclients" {
   match if substring (option vendor-class-identifier, 0, 9) = "PXEClient";
   next-server 10.0.0.1;

   if option architecture-type = 00:07 {
     filename "BOOTX64.efi";
   } else {
     filename "pxelinux/pxelinux.0";
  }
  }
}
```

4. **shim 패키지에서 BOOTX64.efi 파일과 DVD ISO 이미지 파일의 grub2-efi 패키지에서 grubx64.efi 파일에 액세스합니다. 여기서 *my_local_directory* 는 사용자가 생성하는 디렉터리의 이름입니다.**

```
# mount -t iso9660 /path_to_image/name_of_image.iso /mount_point -o loop,ro
```

```
# cp -pr /mount_point/BaseOS/Packages/shim-version-architecture.rpm /my_local_directory
```

```
# cp -pr /mount_point/BaseOS/Packages/grub2-efi-version-architecture.rpm
/my_local_directory
```

```
# umount /mount_point
```

5. **패키지를 추출합니다.**

```
# rpm2cpio shim-version-architecture.rpm | cpio -dimv
```

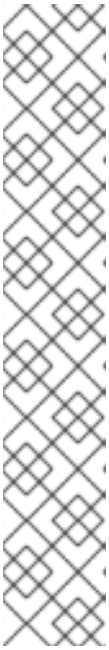
```
# rpm2cpio grub2-efi-version-architecture.rpm | cpio -dimv
```

6. **부팅 디렉터리에서 EFI 부팅 이미지를 복사합니다. *ARCH* 를 shim 또는 grub로 교체한 다음 아키텍처(예: grubx64)를 사용합니다.**

```
# mkdir /var/lib/tftpboot/uefi
# cp my_local_directory/boot/efi/EFI/redhat/ARCH.efi /var/lib/tftpboot/uefi/
```

7. 다음 예와 같이 **grub.cfg**라는 구성 파일을 **tftpboot/** 디렉토리에 추가합니다.

```
set timeout=60
menuentry 'RHEL 8' {
  linuxefi images/RHEL-8.x/vmlinuz ip=dhcp inst.repo=http://10.32.5.1/RHEL-8.x/x86_64/iso-contents-root/
  initrdefi images/RHEL-8.x/initrd.img
}
```



참고

- 설치 프로그램은 런타임 이미지 없이는 부팅할 수 없습니다. **inst.stage2** 부팅 옵션을 사용하여 이미지 위치를 지정합니다. 또는 **inst.repo=** 옵션을 사용하여 이미지와 설치 소스를 지정할 수 있습니다.
- **inst.repo**와 함께 사용되는 설치 소스 위치에는 유효한 **.treeinfo** 파일이 포함되어야 합니다.
- **RHEL8** 설치 **DVD**를 설치 소스로 선택하면 **.treeinfo** 파일은 **BaseOS** 및 **AppStream** 리포지토리를 가리킵니다. 단일 **inst.repo** 옵션을 사용하여 두 리포지토리를 로드할 수 있습니다.

8. **/var/lib/tftpboot/** 디렉터리에 부팅 이미지 파일을 저장하고 부팅 이미지 파일을 디렉터리에 복사합니다. 이 예에서 디렉터리는 **/var/lib/tftpboot/images/RHEL-8.x/** 입니다.

```
# mkdir -p /var/lib/tftpboot/images/RHEL-8/
# cp /path_to_x86_64_images/pxeboot/{vmlinuz,initrd.img} /var/lib/tftpboot/images/RHEL-8/
```

9. **DHCP** 서버에서 **dhcpd** 서비스를 시작하고 활성화합니다. **localhost**에 **DHCP** 서버를 구성한 경우 **localhost**에서 **dhcpd** 서비스를 시작하고 활성화합니다.

```
# systemctl start dhcpd
# systemctl enable dhcpd
```

10. **tftp.socket** 서비스를 시작하고 활성화합니다.

```
# systemctl start tftp.socket
# systemctl enable tftp.socket
```

이제 PXE 부팅 서버가 PXE 클라이언트를 제공할 준비가 되었습니다. Red Hat Enterprise Linux를 설치하는 시스템인 클라이언트를 시작하고, 부팅 소스를 지정하라는 메시지가 표시되면 PXE 부팅 을 선택한 다음 네트워크 설치를 시작할 수 있습니다.

추가 리소스

- Shim 프로그램 사용

14.4. IBM POWER 시스템용 네트워크 서버 구성

GRUB2를 사용하여 IBM Power 시스템의 네트워크 부팅 서버를 구성하려면 다음 절차를 사용하십시오.

중요

이 섹션의 모든 구성 파일은 예시입니다. 구성 세부 정보는 아키텍처 및 특정 요구 사항에 따라 다릅니다.

절차

1. root로 다음 패키지를 설치합니다. 네트워크에 DHCP 서버가 이미 구성되어 있는 경우 dhcp-server 패키지를 제외합니다.

```
# yum install tftp-server dhcp-server
```

2. 방화벽에서 tftp service에 대한 수신 연결을 허용합니다.

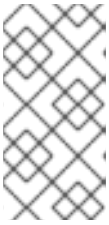
```
# firewall-cmd --add-service=tftp
```

참고

- 이 명령은 다음 서버가 재부팅될 때까지 임시 액세스를 활성화합니다. 영구 액세스를 활성화하려면 명령에 --permanent 옵션을 추가합니다.
- 설치 ISO 파일의 위치에 따라 HTTP 또는 기타 서비스에 대해 들어오는 연결을 허용해야 할 수 있습니다.

3. **tftp 루트 내에 GRUB2 네트워크 부팅 디렉토리를 만듭니다.**

```
# grub2-mknetdir --net-directory=/var/lib/tftpboot
Netboot directory for powerpc-ieee1275 created. Configure your DHCP server to point to
/boot/grub2/powerpc-ieee1275/core.elf
```



참고

명령 출력은 이 절차에 설명된 DHCP 구성에서 구성해야 하는 파일 이름을 알려줍니다.

a. **PXE 서버가 x86 시스템에서 실행되는 경우 tftp 루트 내에 GRUB2 네트워크 부팅 디렉토리를 생성하기 전에 grub2-ppc64-modules 를 설치해야 합니다.**

```
# yum install grub2-ppc64-modules
```

4. **다음 예와 같이 GRUB2 설정 파일 /var/lib/tftpboot/boot/grub2/grub.cfg 를 만듭니다.**

```
set default=0
set timeout=5

echo -e "\nWelcome to the Red Hat Enterprise Linux 8 installer!\n\n"

menuentry 'Red Hat Enterprise Linux 8' {
  linux grub2-ppc64/vmlinuz ro ip=dhcp inst.repo=http://10.32.5.1/RHEL-8/x86_64/iso-
contents-root/
  initrd grub2-ppc64/initrd.img
}
```

참고

- 설치 프로그램은 런타임 이미지 없이는 부팅할 수 없습니다. **inst.stage2** 부팅 옵션을 사용하여 이미지 위치를 지정합니다. 또는 **inst.repo=** 옵션을 사용하여 이미지와 설치 소스를 지정할 수 있습니다.
- **inst.repo**와 함께 사용되는 설치 소스 위치에는 유효한 **.treeinfo** 파일이 포함되어야 합니다.
- **RHEL8** 설치 **DVD**를 설치 소스로 선택하면 **.treeinfo** 파일은 **BaseOS** 및 **AppStream** 리포지토리를 가리킵니다. 단일 **inst.repo** 옵션을 사용하여 두 리포지토리를 로드할 수 있습니다.

5. 명령을 사용하여 **DVD ISO** 이미지를 마운트합니다.

```
# mount -t iso9660 /path_to_image/name_of_iso/ /mount_point -o loop,ro
```

6. 디렉토리를 만들고 **initrd.img** 및 **vmlinuz** 파일을 **DVD ISO** 이미지로 복사합니다. 예를 들면 다음과 같습니다.

```
# cp /mount_point/ppc/ppc64/{initrd.img,vmlinuz} /var/lib/tftpboot/grub2-ppc64/
```

7. 다음 예와 같이 **GRUB2**와 함께 패키지된 부팅 이미지를 사용하도록 **DHCP** 서버를 구성합니다. **DHCP** 서버가 이미 구성된 경우 **DHCP** 서버에서 이 단계를 수행합니다.

```
subnet 192.168.0.1 netmask 255.255.255.0 {
  allow bootp;
  option routers 192.168.0.5;
  group { #BOOTP POWER clients
    filename "boot/grub2/powerpc-ieee1275/core.elf";
    host client1 {
    hardware ethernet 01:23:45:67:89:ab;
    fixed-address 192.168.0.112;
    }
  }
}
```

8. 네트워크 구성에 맞게 샘플 매개변수 서브넷,넷마스크,라우터,**fixed-address** 및 하드웨어 이더넷 을 조정합니다. 파일 이름 매개 변수를 확인합니다. 이 절차의 앞부분에서 **grub2-mknetdir** 명령으로 출력된 파일 이름입니다.

9. **DHCP 서버에서 dhcpd 서비스를 시작하고 활성화합니다. localhost에 DHCP 서버를 구성한 경우 localhost에서 dhcpd 서비스를 시작하고 활성화합니다.**

```
# systemctl start dhcpd
# systemctl enable dhcpd
```

10. **tftp.socket 서비스를 시작하고 활성화합니다.**

```
# systemctl start tftp.socket
# systemctl enable tftp.socket
```

이제 PXE 부팅 서버가 PXE 클라이언트를 제공할 준비가 되었습니다. Red Hat Enterprise Linux를 설치하는 시스템인 클라이언트를 시작하고, 부팅 소스를 지정하라는 메시지가 표시되면 PXE 부팅 을 선택한 다음 네트워크 설치를 시작할 수 있습니다.

15장. 원격 리포지터리 생성

이 절차의 단계에 따라 DVD ISO 이미지의 추출된 콘텐츠가 포함된 원격 리포지토리를 사용하여 네트워크 기반 설치용 설치 소스를 생성합니다. 설치 소스는 HTTP 또는 HTTPS를 통해 액세스할 수 있습니다.

사전 요구 사항

- Red Hat Enterprise Linux 8 설치 DVD/ISO 이미지
- Red Hat Enterprise Linux를 실행하는 여러 서버

15.1. RHEL에 APACHE 설치

이 절차는 Red Hat Enterprise Linux 8에 Apache를 설치하는 데 도움이 됩니다.

사전 요구 사항

- Apache 웹 서버를 사용하여 리포지토리에 액세스

절차

1. httpd 패키지를 설치합니다

```
# yum install httpd
```

2. 실행한 다음 Apache 웹 서버를 활성화합니다. 이러한 명령은 재부팅 후 웹 서버를 시작합니다.

```
# systemctl enable httpd
# systemctl start httpd
```

3. 가지고 있는 웹 사이트 파일을 삽입합니다.

```
# echo Apache on RHEL {ProductNumber} > /var/www/html/index.html
```

4. 방화벽을 업데이트합니다.

```
# firewall-cmd --add-service=http --permanent
# firewall-cmd --add-service=http
```

5. 웹 사이트에 액세스합니다.

```
http://<the-apache-ip-address>

http://<the-apache-hostname>
```

15.2. 원격 리포지터리 생성

여러 Red Hat Enterprise Linux 서버가 네트워크의 단일 Red Hat Enterprise Linux 리포지토리에 액세스할 수 있습니다. 이 경우 실행 중인 웹 서버가 필요합니다. 대부분의 경우 Apache가 됩니다.

사전 요구 사항

- Red Hat Enterprise Linux 8 설치 DVD
- Red Hat Enterprise Linux를 실행하는 여러 서버

절차

1. 다운로드한 DVD의 내용을 마운트 및 복사합니다.

```
mkdir /mnt/rhel{ProductNumber}
mount -o loop,ro rhel-{ProductNumber}-x86_64-dvd.iso /mnt/rhel{ProductNumber}/
cp -r /mnt/rhel{ProductNumber}/ /var/www/html/
umount  /mnt/rhel{ProductNumber}
```

다음 단계는 Apache가 설치된 서버에서 실행되지 않고 클라이언트 측에서 수행됩니다.

2. BaseOS 및 AppStream 리포지토리 모두에 대한 리포지토리 파일을 생성합니다.

```
vi /etc/yum.repos.d/rhel_http_repo.repo

[BaseOS_repo_http]
```

```
name=RHEL_8_x86_64_HTTP BaseOS
baseurl="http://myhost/rhel8/BaseOS"
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release

[AppStream_repo_http]
name=RHEL_8_x86_64_HTTP AppStream
baseurl="http://myhost/rhel8/AppStream"
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release

[root@localhost ~]# yum repolist
Updating Subscription Management repositories.
Unable to read consumer identity
This system is not registered to Red Hat Subscription Management. You can use
subscription-manager to register.
Last metadata expiration check: 0:08:33 ago on Út 23. července 2019, 16:48:09 CEST.
repo id                                          repo name
status
AppStream_repo_http                               RHEL_8_x86_64_HTTP AppStream
4,672
BaseOS_repo_http                                  RHEL_8_x86_64_HTTP BaseOS
1,658
[root@localhost ~]#
```

16장. 부팅 옵션

이 섹션에서는 설치 프로그램의 기본 동작을 수정하는 데 사용할 수 있는 부팅 옵션이 포함되어 있습니다. 전체 부팅 옵션 목록은 업스트림 부팅 옵션 내용을 참조하십시오.

16.1. 부팅 옵션 유형

두 가지 유형의 부팅 옵션은 "=" 기호"와 동일한 " sign"이 없는 부팅 옵션입니다. 부팅 옵션은 부팅 명령줄에 추가되고 공백으로 구분된 여러 옵션을 추가할 수 있습니다. 설치 프로그램과 관련된 부팅 옵션은 항상 inst로 시작합니다.

등호 "=" 기호"가 있는 옵션

= 기호를 사용하는 부팅 옵션의 값을 지정해야 합니다. 예를 들어 inst.vncpassword= 옵션에는 값이 포함되어야 합니다(이 예에서는 암호). 이 예제의 올바른 구문은 inst.vncpassword=password입니다.

동등하지 않은 옵션 "=" sign

이 부팅 옵션은 값 또는 매개변수를 허용하지 않습니다. 예를 들어, rd.live.check 옵션은 설치를 시작하기 전에 설치 미디어를 강제로 설치 프로그램에 강제 적용합니다. 이 부팅 옵션이 있는 경우 설치 프로그램은 확인을 수행하고 부팅 옵션이 없으면 확인을 건너뜁니다.

16.2. 부팅 옵션 편집

이 섹션에는 부팅 메뉴에서 부팅 옵션을 편집할 수 있는 다양한 방법에 대한 정보가 포함되어 있습니다. 설치 미디어를 부팅하면 부팅 메뉴가 열립니다.

16.2.1. BIOS에서 boot: prompt 편집

boot: 프롬프트를 사용하는 경우 첫 번째 옵션은 로드할 설치 프로그램 이미지 파일을 항상 지정해야 합니다. 대부분의 경우 키워드를 사용하여 이미지를 지정할 수 있습니다. 요구 사항에 따라 추가 옵션을 지정할 수 있습니다.

사전 요구 사항

- 부팅 가능한 설치 미디어(USB, CD 또는 DVD)를 생성했습니다.
- 미디어에서 설치를 부팅했으며 설치 부팅 메뉴가 열립니다.

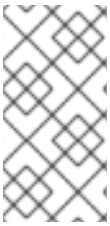
절차

1. 부팅 메뉴를 열고 키보드의 **Esc** 키를 누릅니다.

2. 이제 **boot:** 프롬프트에 액세스할 수 있습니다.

3. 키보드에서 **Tab** 키를 눌러 도움말 명령을 표시합니다.

4. 키보드에서 **Enter** 키를 눌러 옵션으로 설치를 시작합니다. **boot:** 프롬프트에서 부팅 메뉴로 돌아가려면 시스템을 재시작하고 설치 미디어에서 다시 부팅합니다.



참고

boot: prompt는 **dracut** 커널 옵션도 허용합니다. 옵션 목록은 **dracut.cmdline(7)** 매뉴얼 페이지에서 사용할 수 있습니다.

16.2.2. > 프롬프트를 사용하여 사전 정의된 부팅 옵션 편집

BIOS 기반 **AMD64** 및 **Intel 64** 시스템에서는 **>** 프롬프트를 사용하여 사전 정의된 부팅 옵션을 편집할 수 있습니다. 전체 옵션 세트를 표시하려면 이 미디어 테스트를 선택하고 부팅 메뉴에서 **RHEL 8**을 설치합니다.

사전 요구 사항

- 부팅 가능한 설치 미디어(**USB, CD** 또는 **DVD**)를 생성했습니다.
- 미디어에서 설치를 부팅했으며 설치 부팅 메뉴가 열립니다.

절차

1. 부팅 메뉴에서 옵션을 선택하고 키보드의 **Tab** 키를 누릅니다. **& gt;** 프롬프트에 액세스할 수 있으며 사용 가능한 옵션을 표시합니다.

2. 필요한 옵션을 **>** 프롬프트에 추가합니다.

3. Enter를 눌러 설치를 시작합니다.

4. Esc 를 눌러 편집을 취소하고 부팅 메뉴로 돌아갑니다.

16.2.3. UEFI 기반 시스템의 GRUB2 메뉴 편집

GRUB2 메뉴는 UEFI 기반 AMD64, Intel 64 및 64비트 ARM 시스템에서 사용할 수 있습니다.

사전 요구 사항

- 부팅 가능한 설치 미디어(USB, CD 또는 DVD)를 생성했습니다.
- 미디어에서 설치를 부팅했으며 설치 부팅 메뉴가 열립니다.

절차

1. 부팅 메뉴 창에서 필요한 옵션을 선택하고 e 를 누릅니다.

2. UEFI 시스템에서 커널 명령줄은 linuxefi로 시작합니다. 커서를 linuxefi 커널 명령줄의 끝으로 이동합니다.

3. 필요에 따라 매개 변수를 편집합니다. 예를 들어 하나 이상의 네트워크 인터페이스를 구성하려면 linuxefi 커널 명령줄 끝에 ip= 매개 변수를 추가한 다음 필수 값을 추가합니다.

4. 편집을 마치면 Ctrl+X 를 눌러 지정된 옵션을 사용하여 설치를 시작합니다.

16.3. 설치 소스 부팅 옵션

이 섹션에서는 다양한 설치 소스 부팅 옵션에 대해 설명합니다.

inst.repo=

inst.repo= boot 옵션은 설치 소스, 즉 패키지 리포지토리를 제공하는 위치와 이를 설명하는 유효한 .treeinfo 파일을 지정합니다. 예: inst.repo=cdrom. inst.repo= 옵션의 대상은 다음 설치 미디어 중

하나여야 합니다.

- 설치 프로그램 이미지, 패키지 및 리포지토리 데이터와 유효한 **.treeinfo** 파일을 포함하는 디렉터리 구조인 설치 가능 트리

- **DVD** (시스템 **DVD** 드라이브에 물리적 디스크가 있음)

- 전체 **Red Hat Enterprise Linux** 설치 **DVD**의 **ISO** 이미지는 하드 드라이브 또는 시스템에서 액세스할 수 있는 네트워크 위치에 배치됩니다.

inst.repo= 부트 옵션을 사용하여 다른 형식을 사용하여 다양한 설치 방법을 구성합니다. 다음 표에는 **inst.repo=** 부트 옵션 구문에 대한 세부 정보가 포함되어 있습니다.

표 16.1. inst.repo= 부팅 옵션 및 설치 소스의 유형 및 형식

소스 유형	부팅 옵션 형식	소스 형식
CD/DVD 드라이브	**inst.repo=cdrom:*<device>***	DVD를 물리적 디스크로 설치 [a]
마운트 가능한 장치 (HDD 및 USB 고착)	**inst.repo=hd:*<device>*:/*<path>***	설치 DVD의 이미지 파일입니다.
NFS 서버	**inst.repo=nfs:[*options*:]*<server>*:/*<path>***	설치 DVD의 이미지 파일 또는 설치 DVD에 있는 디렉터리 및 파일의 전체 사본인 설치 트리입니다. [b]
HTTP 서버	**inst.repo=http://*<host>*/*<path>***	설치 DVD에서 디렉터리 및 파일의 전체 사본인 설치 트리.
HTTPS 서버	**inst.repo=https://*<host>*/*<path>***	
FTP 서버	**inst.repo=ftp://*<username>*:*<password>*@*<host>*/*<path>***	
HMC	**inst.repo=hmc**	

[a] *장치가 꺼져* 있으면 설치 프로그램은 설치 DVD가 포함된 드라이브를 자동으로 검색합니다.

[b] NFS Server 옵션은 기본적으로 NFS 프로토콜 버전 3을 사용합니다. 다른 버전을 사용하려면 **nfsvers=*X*** 를 *options*에 추가하고 *X*를 사용하려는 버전 번호로 대체합니다.

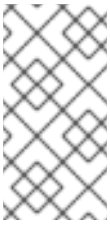
다음과 같은 형식으로 디스크 장치 이름을 설정합니다.

- 커널 장치 이름 (예: /dev/sda1 또는 sdb2)
- 파일 시스템 레이블 (예: LABEL= skopeo 또는 LABEL=RHEL8)
- 파일 시스템 UUID (예: UUID=8176c7bf-04ff-403a-a832-9557f94e61db)

영숫자가 아닌 문자는 \xNN으로 표시되어야 합니다. 여기서 *NN*은 문자의 16진수 표현입니다. 예를 들어 \x20은 공백 (" ") 입니다.

inst.addrepo=

inst.addrepo= boot 옵션을 사용하여 기본 리포지토리(inst.repo=)와 함께 다른 설치 소스로 사용할 수 있는 추가 리포지토리를 추가합니다. 부팅 중에 inst.addrepo= 부트 옵션을 여러 번 사용할 수 있습니다. 다음 표에는 inst.addrepo= 부팅 옵션 구문에 대한 세부 정보가 포함되어 있습니다.



참고

REPO_NAME은 리포지토리의 이름이며 설치 프로세스에 필요합니다. 이러한 리포지토리는 설치 프로세스 중에만 사용되며 설치된 시스템에 설치되지 않습니다.

통합 ISO에 대한 자세한 내용은 통합 ISO를 참조하십시오.

표 16.2. 설치 소스 및 부팅 옵션 형식

설치 소스	부팅 옵션 형식	추가 정보
URL에 설치 가능한 트리	inst.addrepo=REPO_NAME,[http,https,ftp]://*<host>*/*<path>*	지정된 URL에서 설치 가능한 트리를 찾습니다.
NFS 경로에 설치 가능한 트리	inst.addrepo=REPO_NAME,nfs://*<server>*:/*<path>*	지정된 NFS 경로에서 설치 가능한 트리를 찾습니다. 호스트 다음에 콜론이 필요합니다. 설치 프로그램은 RFC 2224에 따른 URL 구문 분석 대신 nfs:// 이후의 모든 항목을 마운트 명령에 직접 전달합니다.

설치 소스	부팅 옵션 형식	추가 정보
설치 환경에 설치 가능한 트리	**inst.addrepo=REPO_NAME,file:///*<path>***	설치 환경의 지정된 위치에서 설치할 수 있는 트리를 찾습니다. 이 옵션을 사용하려면 설치 프로그램에서 사용 가능한 소프트웨어 그룹을 로드하기 전에 리포지토리를 마운트해야 합니다. 이 옵션을 사용하면 하나의 부팅 가능 ISO에 여러 리포지토리를 사용할 수 있으며, ISO에서 기본 리포지토리와 추가 리포지토리를 모두 설치할 수 있습니다. 추가 리포지토리의 경로는 **/run/install/source/REPO_ISO_PATH**입니다. 또한 Kickstart 파일의 **%pre** 섹션에 리포지토리 디렉터리를 마운트할 수 있습니다. 경로는 절대적이어야 하며 /로 시작해야 합니다. (예: **inst.addrepo=REPO_NAME,file:///*<path>***)
하드 드라이브	**inst.addrepo=REPO_NAME,hd:*<device>*:*<path>***	지정된 *<device>* 파티션을 마운트하고 *<path>*에 지정된 ISO에서 설치합니다. *<path>*를 지정하지 않으면 설치 프로그램은 *<device>*에서 유효한 설치 ISO를 찾습니다. 이 설치 방법을 사용하려면 설치 가능한 유효한 트리가 있는 ISO가 필요합니다.

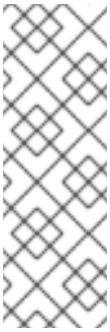
inst.stage2=

inst.stage2= 부트 옵션은 설치 프로그램의 런타임 이미지의 위치를 지정합니다. 이 옵션은 유효한 .treeinfo 파일이 포함된 디렉터리의 경로를 예상하고 .treeinfo 파일에서 런타임 이미지 위치를 읽습니다. .treeinfo 파일을 사용할 수 없는 경우 설치 프로그램은 images/install.img 에서 이미지를 로드하려고 합니다.

inst.stage2 옵션을 지정하지 않으면 설치 프로그램에서 inst.repo 옵션으로 지정된 위치를 사용하려고 합니다.

나중에 설치 프로그램에서 설치 소스를 수동으로 지정하려면 이 옵션을 사용합니다. 예를 들어, CDN(Content Delivery Network)을 설치 소스로 선택하려는 경우입니다. 설치 DVD 및 부팅 ISO에는 각 ISO에서 설치 프로그램을 부팅하는 데 적합한 inst.stage2 옵션이 이미 포함되어 있습니다.

설치 소스를 지정하려면 대신 inst.repo= 옵션을 사용합니다.



참고

기본적으로 **inst.stage2=** 부트 옵션은 설치 미디어에서 사용되며 특정 레이블로 설정됩니다(예: **inst.stage2=hd:LABEL=RHEL-x-0-BaseOS-x86_64**). 런타임 이미지가 포함된 파일 시스템의 기본 레이블을 수정하거나 사용자 지정 프로시저를 사용하여 설치 시스템을 부팅하는 경우 **inst.stage2=** 부팅 옵션이 올바른 값으로 설정되어 있는지 확인합니다.

inst.noverifyssl

inst.noverifyssl 부팅 옵션을 사용하여 설치 프로그램이 추가 **Kickstart** 리포지토리를 제외하고 모든 **HTTPS** 연결에 대한 **SSL** 인증서를 확인하지 않도록 합니다. 여기서 **--noverifyssl**을 리포지토리별로 설정할 수 있습니다.

예를 들어 원격 설치 소스가 자체 서명된 **SSL** 인증서를 사용하는 경우 **inst.noverifyssl** 부팅 옵션을 사용하면 설치 프로그램이 **SSL** 인증서를 확인하지 않고 설치를 완료할 수 있습니다.

inst.stage2=를 사용하여 소스를 지정할 때의 예

```
inst.stage2=https://hostname/path_to_install_image/ inst.noverifyssl
```

inst.repo=를 사용하여 소스를 지정할 때의 예

```
inst.repo=https://hostname/path_to_install_repository/ inst.noverifyssl
```

inst.stage2.all

inst.stage2.all 부팅 옵션을 사용하여 여러 **HTTP, HTTPS** 또는 **FTP** 소스를 지정합니다. **inst.stage2=** 부팅 옵션을 **inst.stage2.all** 옵션과 함께 여러 번 사용하여 성공할 때까지 소스에서 이미지를 순차적으로 가져올 수 있습니다. 예를 들어 다음과 같습니다.

```
inst.stage2.all
inst.stage2=http://hostname1/path_to_install_tree/
inst.stage2=http://hostname2/path_to_install_tree/
inst.stage2=http://hostname3/path_to_install_tree/
```

inst.dd=

inst.dd= 부팅 옵션은 설치 중에 드라이버 업데이트를 수행하는 데 사용됩니다. 설치 중에 드라이버를 업데이트하는 방법에 대한 자세한 내용은 *고급 RHEL 8 설치* 문서를 참조하십시오.

inst.repo=hmc

이 옵션은 외부 네트워크 설정 요구 사항을 제거하고 설치 옵션을 확장합니다. 바이너리 **DVD**에서 부팅할 때 설치 프로그램에서 추가 커널 매개 변수를 입력하라는 메시지를 표시합니다. **DVD**를 설치 소스로 설정하려면 **inst.repo=hmc** 옵션을 커널 매개 변수에 추가합니다. 그러면 설치 프로그램에서 지원 요소(**SE**) 및 하드웨어 관리 콘솔(**HMC**) 파일 액세스를 활성화하고, **DVD**에서 **stage2** 이미지를 가져온 다음 소프트웨어 선택을 위해 **DVD**의 패키지에 액세스할 수 있습니다.

inst.proxy=

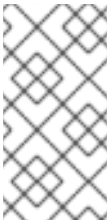
inst.proxy= 부팅 옵션은 **HTTP**, **HTTPS** 및 **FTP** 프로토콜에서 설치를 수행할 때 사용됩니다. 예를 들어 다음과 같습니다.

```
[PROTOCOL://][USERNAME[:PASSWORD]@]HOST[:PORT]
```

inst.nosave=

inst.nosave= boot 옵션을 사용하여 설치된 시스템에 저장되지 않은 설치 로그 및 관련 파일(예: **input_ks,output_ks,all_ks,logs** 및 **all**)을 제어합니다. 여러 값을 쉼표로 구분하여 결합할 수 있습니다. 예를 들면 다음과 같습니다.

```
inst.nosave=Input_ks,logs
```



참고

inst.nosave 부팅 옵션은 로그 및 입력/출력 **Kickstart** 결과와 같은 **Kickstart** **%post** 스크립트에서 제거할 수 없는 설치된 시스템의 파일을 제외하는 데 사용됩니다.

input_ks

입력 **Kickstart** 결과를 저장하는 기능을 비활성화합니다.

output_ks

설치 프로그램에서 생성된 출력 **Kickstart** 결과를 저장하는 기능을 비활성화합니다.

all_ks

입력 및 출력 **Kickstart** 결과를 저장하는 기능을 비활성화합니다.

logs

모든 설치 로그를 저장하는 기능을 비활성화합니다.

all

모든 **Kickstart** 결과 및 모든 로그를 저장하는 기능을 비활성화합니다.

inst.multilib

inst.multilib 부팅 옵션을 사용하여 **DNF**의 **multilib_policy**를 **best**가 아닌 **all**로 설정합니다.

inst.memcheck

inst.memcheck 부팅 옵션은 설치를 완료하는 데 충분한 **RAM**이 있는지 확인하는 검사를 수행합니다. **RAM**이 충분하지 않으면 설치 프로세스가 중지됩니다. 설치 중에 시스템 점검은 대략적이고 메모리 사용은 패키지 선택, 사용자 인터페이스(예: 그래픽 또는 텍스트) 및 기타 매개 변수에 따라 달라집니다.

inst.nomemcheck

inst.nomemcheck 부팅 옵션은 설치를 완료하기에 충분한 **RAM**이 있는지 확인하기 위해 검사를 수행하지 않습니다. 최소 메모리 용량이 권장되지 않는 메모리보다 적은 설치를 시도하면 설치 프로세스가 실패할 수 있습니다.

16.4. 네트워크 부팅 옵션

로컬 이미지에서 부팅하지 않고 네트워크를 통해 이미지를 부팅해야 하는 경우 다음 옵션을 사용하여 네트워크 부팅을 사용자 지정할 수 있습니다.

참고

dracut 툴을 사용하여 네트워크를 초기화합니다. **dracut** 옵션 전체 목록은 **dracut.cmdline(7)** 매뉴얼 페이지를 참조하십시오.

ip=

ip= boot 옵션을 사용하여 하나 이상의 네트워크 인터페이스를 구성합니다. 여러 인터페이스를 구성하려면 다음 방법 중 하나를 사용합니다.

- 각 인터페이스에 한 번 **ip** 옵션을 여러 번 사용합니다. 이렇게 하려면 **rd.neednet=1** 옵션을 사용하고 **bootdev** 옵션을 사용하여 기본 부팅 인터페이스를 지정합니다.

- **ip** 옵션을 한 번 사용한 다음 **Kickstart**를 사용하여 추가 인터페이스를 설정합니다. 이 옵션에는 여러 다른 형식을 사용할 수 있습니다. 다음 테이블에는 가장 일반적인 옵션에 대한 정보가 포함되어 있습니다.

다음 표에서는 다음을 수행합니다.

- **ip** 매개 변수는 클라이언트 **IP** 주소를 지정하고 대괄호(예: **[2001:db8::99]**)가 필요합니다.
- **gateway** 매개 변수는 기본 게이트웨이입니다. **IPv6** 주소도 허용됩니다.
- **netmask** 매개 변수는 사용할 넷마스크입니다. 전체 넷마스크(예: **255.255.255.0**) 또는 접두사(예: **64**)일 수 있습니다.
- **hostname** 매개 변수는 클라이언트 시스템의 호스트 이름입니다. 이 매개변수는 선택 사항입니다.

표 16.3. 네트워크 인터페이스를 구성하기 위한 부팅 옵션 형식

부팅 옵션 형식	구성 방법
ip=*method*	인터페이스 자동 설정
ip=*interface:method*	특정 인터페이스의 자동 설정
ip=*ip::gateway:netmask:hostname:interface*:none	정적 구성
ip=*ip::gateway:netmask:hostname:interface:method:mtu*	덮어쓰기를 사용하는 특정 인터페이스의 자동 구성

자동 인터페이스에 대한 설정 방법

재정의를 사용하여 특정 인터페이스의 자동 구성은 **dhcp** 와 같은 지정된 자동 구성 방법을 사용하여 인터페이스를 엽니다. 그러나 자동으로 가져온 **IP** 주소, 게이트웨이, 넷마스크, 호스트 이름 또는 기타 지정된 매개 변수를 덮어씁니다. 모든 매개변수는 선택 사항이므로 재정의할 매개변수만 지정합니다.

method 매개변수는 다음 중 하나일 수 있습니다.

DHCP

dhcp

IPv6 DHCP

dhcp6

IPv6 자동 구성

auto6

iSCSI 부팅 펌웨어 테이블**(iBFT)**

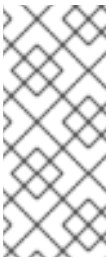
ibft

참고

- **ip** 옵션을 지정하지 않고 **inst.ks=http://host/path** 와 같은 네트워크 액세스 권한이 필요한 부팅 옵션을 사용하는 경우 **ip =dhcp..**
- **iSCSI** 대상에 자동으로 연결하려면 **ip=ibft** 부팅 옵션을 사용하여 대상에 액세스하기 위해 네트워크 장치를 활성화합니다.

nameserver=

nameserver= 옵션은 이름 서버의 주소를 지정합니다. 이 옵션을 여러 번 사용할 수 있습니다.



참고

ip= 매개 변수에는 대괄호가 필요합니다. 그러나 **IPv6** 주소는 대괄호로 묶지 않습니다. **IPv6** 주소에 사용할 올바른 구문의 예는 **nameserver=2001:db8::1**입니다.

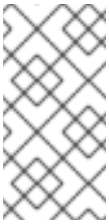
bootdev=

bootdev= 옵션은 부팅 인터페이스를 지정합니다. **ip** 옵션을 두 개 이상 사용하는 경우 이 옵션이 필요합니다.

ifname=

ifname= 옵션은 지정된 **MAC** 주소가 있는 네트워크 장치에 인터페이스 이름을 할당합니다. 이 옵션을 여러 번 사용할 수 있습니다. 구문은 **ifname=interface:MAC**입니다. 예를 들어 다음과 같습니다.

```
ifname=eth0:01:23:45:67:89:ab
```



참고

ifname= 옵션은 설치 중에 사용자 지정 네트워크 인터페이스 이름을 설정하는 유일한 방법입니다.

inst.dhcpclass=

inst.dhcpclass= 옵션은 **DHCP** 공급업체 클래스 식별자를 지정합니다. **dhcpd** 서비스는 이 값을 **vendor-class-identifier**로 확인합니다. 기본값은 **anaconda-$(uname -srm)**입니다.

inst.waitfornet=

inst.waitfornet=SECONDS 부팅 옵션을 사용하면 설치 시스템이 설치 전에 네트워크 연결을 기다릴 수 있습니다. **SECONDS** 인수에서 제공되는 값은 시간 초과 전에 네트워크 연결을 대기하고 네트워크 연결이 존재하지 않는 경우에도 설치 프로세스를 계속할 때까지 대기하는 최대 시간을 지정합니다.

vlan=

vlan= 옵션을 사용하여 지정된 이름의 지정된 인터페이스에서 **VLAN(Virtual LAN)** 장치를 구성합니다. 구문은 **vlan=name:interface**입니다. 예를 들어 다음과 같습니다.

```
vlan=vlan5:enp0s1
```

이렇게 하면 **enp0s1** 인터페이스에서 **vlan5** 라는 **VLAN** 장치가 구성됩니다. 이름은 다음 형식을 사용할 수 있습니다.

- **VLAN_PLUS_VID: vlan0005**
- **VLAN_PLUS_VID_NO_PAD: vlan5**
- **DEV_PLUS_VID: enp0s1.0005**

- **DEV_PLUS_VID_NO_PAD: enp0s1.5**

bond=

bond= 옵션을 사용하여 **bond=name[:interfaces][:options]** 구문을 사용하여 본딩 장치를 구성합니다. ***name***을 본딩 장치 이름으로, *인터페이스*를 쉼표로 구분된 물리**(Ethernet)** 인터페이스로, *옵션을* 쉼표로 구분된 본딩 옵션 목록으로 교체합니다. 예를 들어 다음과 같습니다.

```
bond=bond0:enp0s1,enp0s2:mode=active-backup,tx_queues=32,downdelay=5000
```

사용 가능한 옵션 목록을 보려면 **modinfo** 본딩 명령을 실행합니다.

team=

team= 옵션을 사용하여 **team=name:interfaces** 구문으로 팀 장치를 구성합니다. ***name*** 을 팀 장치의 원하는 이름으로 바꾸고, 팀 장치에서 기본 인터페이스로 사용할 이름을 쉼표로 구분된 물리적**(Ethernet)** 장치로 바꿉니다. *예를 들어 다음과 같습니다.*

```
team=team0:enp0s1,enp0s2
```

bridge=

bridge= *옵션을 사용하여* ***bridge=name:interfaces*** *구문을 사용하여 브리지 장치를 구성합니다.* ***name****을 브리지 장치 및* ***interfaces****의 원하는 이름으로 교체하고, 브릿지 장치에서 기본 인터페이스로 사용할 물리적****(Ethernet)*** *장치 목록을 쉼표로 구분한 목록으로 바꿉니다. 예를 들어 다음과 같습니다.*

```
bridge=bridge0:enp0s1,enp0s2
```

추가 리소스

- *네트워킹 구성 및 관리*

16.5. 콘솔 부팅 옵션

이 섹션에서는 콘솔의 부팅 옵션, 모니터 디스플레이 및 키보드를 구성하는 방법을 설명합니다.

console=

console= *옵션을 사용하여 기본 콘솔로 사용할 장치를 지정합니다. 예를 들어 첫 번째 직렬 포트*

에서 콘솔을 사용하려면 **console=ttyS0**을 사용합니다. **console=** 인수를 사용하면 텍스트 **UI**로 설치가 시작됩니다. **console=** 옵션을 여러 번 사용해야 하는 경우 지정된 모든 콘솔에 부팅 메시지가 표시됩니다. 그러나 설치 프로그램은 마지막 지정된 콘솔만 사용합니다. 예를 들어 **console=ttyS0 console=ttyS1**을 지정하면 설치 프로그램에서 **ttyS1**을 사용합니다.

inst.lang=

설치 중에 사용할 언어를 설정하려면 **inst.lang=** 옵션을 사용합니다. 로케일 목록을 보려면 **locale -a | grep _** 또는 **localectl list-locales | grep _** 명령을 입력합니다.

inst.singlelang

inst.singlelang 옵션을 사용하여 단일 언어 모드로 설치하면 설치 언어 및 언어 지원 구성에 사용 가능한 대화형 옵션이 없습니다. **inst.lang** 부팅 옵션 또는 **lang Kickstart** 명령을 사용하여 언어를 지정하는 경우 해당 언어가 사용됩니다. 언어를 지정하지 않으면 설치 프로그램의 기본값은 **en_US.UTF-8** 입니다.

inst.geoloc=

설치 프로그램에서 **inst.geoloc=** 옵션을 사용하여 위치 사용을 구성합니다. **Geolocation**은 언어 및 시간대를 사전 설정하는 데 사용되며 **inst.geoloc=value** 구문을 사용합니다. **value**는 다음 매개변수 중 하나일 수 있습니다.

- **geolocation: inst.geoloc=0**을 비활성화합니다.
- **Fedora GeoIP API** 사용**: inst.geoloc=provider_fedora_geoip**
- **Hostip.info GeoIP API** 사용**: inst.geoloc=provider_hostip**

inst.geoloc= 옵션을 지정하지 않으면 기본 옵션은 **provider_fedora_geoip** 입니다.

inst.keymap=

inst.keymap= 옵션을 사용하여 설치에 사용할 키보드 레이아웃을 지정합니다.

inst.cmdline

inst.cmdline 옵션을 사용하여 설치 프로그램이 명령줄 모드에서 실행되도록 합니다. 이 모드는 상호 작용을 허용하지 않으며 **Kickstart** 파일 또는 명령줄의 모든 옵션을 지정해야 합니다.

inst.graphical

inst.graphical 옵션을 사용하여 설치 프로그램이 그래픽 모드에서 실행되도록 합니다. 그래픽 모

드가 기본값입니다.

inst.text

inst.text 옵션을 사용하여 설치 프로그램이 그래픽 모드 대신 텍스트 모드에서 실행되도록 합니다.

inst.noninteractive

비대화형 모드에서 설치 프로그램을 실행하려면 **inst.noninteractive** 부팅 옵션을 사용합니다. 비대화형 모드에서는 사용자 상호 작용이 허용되지 않으며, **inst.nointeractive** 옵션을 그래픽 또는 텍스트 설치와 함께 사용할 수 있습니다. 텍스트 모드에서 **inst.noninteractive** 옵션을 사용하면 **inst.cmdline** 옵션과 동일하게 작동합니다.

inst.resolution=

inst.resolution= 옵션을 사용하여 그래픽 모드에서 화면 해상도를 지정합니다. 형식은 **NxM** 입니다. 여기서 **N**은 화면 너비이고 **M**은 화면 높이(**px**)입니다. 지원되는 가장 낮은 해상도는 **1024x768**입니다.

inst.vnc

inst.vnc 옵션을 사용하여 **VNC(Virtual Network Computing)**를 사용하여 그래픽 설치를 실행합니다. 설치 프로그램과 상호 작용하려면 **VNC** 클라이언트 애플리케이션을 사용해야 합니다. **VNC** 공유가 활성화되면 여러 클라이언트가 연결할 수 있습니다. **VNC**를 사용하여 설치된 시스템은 텍스트 모드에서 시작됩니다.

inst.vncpassword=

설치 프로그램에서 사용하는 **VNC** 서버에서 암호를 설정하려면 **inst.vncpassword=** 옵션을 사용합니다.

inst.vncconnect=

inst.vncconnect= 옵션을 사용하여 지정된 호스트 위치에서 수신 대기 중인 **VNC** 클라이언트에 연결합니다(예: **inst.vncconnect=<host>[:<port>]** 기본 포트는 **5900**입니다. **vncviewer -listen** 명령을 입력하여 이 옵션을 사용할 수 있습니다.

inst.xdriver=

inst.xdriver= 옵션을 사용하여 설치 중 및 설치된 시스템에서 모두 사용할 **X** 드라이버 이름을 지정합니다.

inst.usefbx

inst.usefbx 옵션을 사용하여 하드웨어별 드라이버 대신 프레임 버퍼 **X** 드라이버를 사용하도록 설치 프로그램에 메시지를 표시합니다. 이 옵션은 **inst.xdriver=fbdev** 옵션과 동일합니다.

modprobe.blacklist=

modprobe.blacklist= 옵션을 사용하여 하나 이상의 드라이버를 차단 목록에 표시하거나 완전히 비활성화합니다. 이 옵션을 사용하지 않도록 설정하는 드라이버(**mods**)는 설치가 시작될 때 로드할 수 없습니다. 설치가 완료되면 설치된 시스템에서 이러한 설정을 유지합니다. **blocklisted** 드라이버 목록은 **/etc/modprobe.d/** 디렉토리에서 찾을 수 있습니다. 쉼표로 구분된 목록을 사용하여 여러 드라이버를 비활성화합니다. 예를 들어 다음과 같습니다.

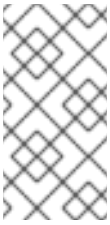
```
modprobe.blacklist=ahci,firewire_ohci
```

inst.xtimeout=

inst.xtimeout= 옵션을 사용하여 **X** 서버를 시작하는 데 시간 초과를 초 단위로 지정합니다.

inst.sshd

SSH를 사용하여 시스템에 연결하고 설치 진행 상황을 모니터링할 수 있도록 **inst.sshd** 옵션을 사용하여 설치 중에 **sshd** 서비스를 시작합니다. **SSH**에 대한 자세한 내용은 **ssh(1)** 도움말 페이지를 참조하십시오. 기본적으로 **sshd** 옵션은 **64**비트 **IBM Z** 아키텍처에서만 자동으로 시작됩니다. 다른 아키텍처에서 **inst.sshd** 옵션을 사용하지 않는 경우 **sshd**가 시작되지 않습니다.



참고

설치하는 동안 **root** 계정에는 기본적으로 암호가 없습니다. **sshpw Kickstart** 명령을 사용하여 설치 중에 **root** 암호를 설정할 수 있습니다.

inst.kdump_addon=

inst.kdump_addon= 옵션을 사용하여 설치 프로그램에서 **Kdump** 구성 화면(**add-on**)을 활성화하거나 비활성화합니다. 이 화면은 기본적으로 활성화되어 있습니다. **inst.kdump_addon=off**를 사용하여 비활성화합니다. 애드온을 비활성화하면 그래픽 및 텍스트 기반 인터페이스의 **Kdump** 화면과 **%addon com_redhat_kdump Kickstart** 명령이 비활성화됩니다.

16.6. 디버그 부팅 옵션

이 섹션에서는 문제를 디버깅할 때 사용할 수 있는 옵션에 대해 설명합니다.

inst.rescue

inst.rescue 옵션을 사용하여 시스템 진단 및 수정에 필요한 복구 환경을 실행합니다. 예를 들어 복구 모드에서 파일 시스템을 복구할 수 있습니다.

inst.updates=

inst.updates= 옵션을 사용하여 설치 중에 적용할 **updates.img** 파일의 위치를 지정합니다. **updates.img** 파일은 여러 소스 중 하나에서 파생될 수 있습니다.

표 16.4. updates.img 파일 소스

소스	설명	예제
네트워크에서 업데이트	**updates.img** 의 네트워크 위치를 지정합니다. 설치 트리를 수정할 필요가 없습니다. 이 방법을 사용하려면 **inst.updates**를 포함하도록 커널 명령행을 편집합니다.	**inst.updates=http://website.com/path/to/updates.img**.
디스크 이미지에서 업데이트	플로피 드라이브 또는 USB 키에 **updates.img** 를 저장합니다. 이는 **ext2** 파일 시스템 유형의 **updates.img**에서만 수행할 수 있습니다. 이미지 내용을 플로피 드라이브에 저장하려면 플로피 디스크를 삽입하고 명령을 실행합니다.	**DD if=updates.img of=/dev/fd0 bs=72k count=20**. USB 키 또는 플래쉬 미디어를 사용하려면 **/dev/fd0** 을 USB 플래시 드라이브의 장치 이름으로 교체합니다.
설치 트리에서 업데이트	CD, 하드 드라이브, HTTP 또는 FTP 설치를 사용하는 경우 모든 설치가 **.img** 파일을 감지할 수 있도록 **updates.img** 를 설치 트리에 저장합니다. 파일 이름은 **updates.img**여야 합니다.	NFS 설치의 경우 파일을 **images/** 디렉터리에 저장하거나 **RHupdates/** 디렉터리에 저장합니다.

inst.loglevel=

inst.loglevel= 옵션을 사용하여 터미널에 기록된 최소 메시지 수준을 지정합니다. 이 옵션은 터미널 로깅에만 적용됩니다. 로그 파일에는 항상 모든 수준의 메시지가 포함됩니다. 이 옵션에 대한 가능한 값은 가장 낮은 수준의 최고 수준입니다.

- ***debug***
- ***info***
- ***경고***
- ***error***

- *심각*

*기본값은 **info** 입니다. 즉, 기본적으로 로깅 터미널은 **info** 에서 중요한 까지의 메시지를 표시합니다.*

inst.syslog=

*설치가 시작될 때 지정된 호스트의 **syslog** 프로세스로 로그 메시지를 보냅니다. 들어오는 연결을 허용하도록 원격 **syslog** 프로세스가 구성된 경우에만 **inst.syslog=** 를 사용할 수 있습니다.*

inst.virtiolog=

*__inst.virtiolog=__ 옵션을 사용하여 로그 전달에 사용할 **virtio** 포트(**/dev/virtio-ports/name**에서 문자 장치)를 지정합니다. 기본값은 **org.fedoraproject.anaconda.log.0** 입니다.*

inst.zram=

*설치 중에 **zRAM** 스왑의 사용을 제어합니다. 옵션은 시스템 **RAM** 내에 압축된 블록 장치를 만들고 하드 드라이브를 사용하는 대신 스왑 공간에 사용합니다. 이 설정을 사용하면 설치 프로그램을 사용 가능한 메모리로 실행하고 설치 속도를 향상시킬 수 있습니다. 다음 값을 사용하여 **inst.zram=** 옵션을 구성할 수 있습니다.*

- *__inst.zram=1__에서 시스템 메모리 크기에 관계없이 **zRAM** 스왑을 활성화합니다. 기본적으로 **zRAM**의 스왑은 **2GiB** 이하의 **RAM**이 있는 시스템에서 활성화됩니다.*

- *__inst.zram=0__에서 시스템 메모리 크기에 관계없이 **zRAM** 스왑을 비활성화합니다. 기본적으로 **zRAM**의 스왑은 **2GiB** 이상의 메모리가 있는 시스템에서 비활성화되어 있습니다.*

rd.live.ram

*__images/install.img__ 의 2단계 이미지를 **RAM**에 복사합니다. 이렇게 하면 이미지 크기에 필요한 메모리가 증가하며 일반적으로 **400~800MB** 사이입니다.*

inst.nokill

치명적인 오류가 발생하거나 설치 프로세스가 끝나면 설치 프로그램이 재부팅되지 않도록 합니다. 재부팅 시 손실되는 설치 로그를 캡처하여 사용합니다.

inst.noshell

*설치하는 동안 터미널 세션 **2(tty2)**의 쉘을 방지합니다.*

inst.notmux

*설치 중에 **tmux**를 사용하지 않도록 합니다. 출력은 터미널 제어 문자 없이 생성되며 대화형이 아닌 용도로 사용됩니다.*

inst.remotelog=

***TCP** 연결을 사용하여 모든 로그를 원격 호스트**:port** 로 보냅니다. 리스너가 없고 설치가 정상적으로 진행되면 연결이 중단됩니다.*

16.7. 스토리지 부팅 옵션

이 섹션에서는 스토리지 장치에서 부팅을 사용자 지정하기 위해 지정할 수 있는 옵션에 대해 설명합니다.

inst.nodmraid

***dmraid** 지원을 비활성화합니다.*

주의

*이 옵션을 주의해서 사용하십시오. 펌웨어 **RAID** 배열의 일부로 잘못 식별되는 디스크가 있는 경우 **dmraid** 또는 **wipefs** 와 같은 적절한 도구를 사용하여 제거해야 하는 오래된 **RAID** 메타데이터가 있을 수 있습니다.*

inst.nompath

*다중 경로 장치 지원을 비활성화합니다. 일반 블록 장치를 다중 경로 장치로 잘못 식별하는 **false-positive**가 있는 경우에만 이 옵션을 사용합니다.*

주의

이 옵션을 주의해서 사용하십시오. 다중 경로 하드웨어에 이 옵션을 사용하지 마십시오. 이 옵션을 사용하여 다중 경로 장치의 단일 경로에 설치하는 것은 지원되지 않습니다.

inst.gpt

설치 프로그램이 **MBR(Master Boot Record)**이 아닌 **GUID** 파티션 테이블(**GPT**)에 파티션 정보를 설치하도록 강제 적용합니다. 이 옵션은 **BIOS** 호환성 모드에 있지 않는 한 **UEFI** 기반 시스템에서 유효하지 않습니다. 일반적으로 **BIOS** 기반 시스템 및 **BIOS** 호환성 모드의 **UEFI** 기반 시스템은 디스크 크기가 **2^32** 섹터이거나 큰 경우 파티션 정보를 저장하기 위해 **DASD** 스키마를 사용하려고 합니다. 디스크 섹터는 일반적으로 크기가 **512**바이트이므로 일반적으로 **2TiB**와 동일합니다. **inst.gpt** 부팅 옵션을 사용하면 **GPT**를 더 작은 디스크에 쓸 수 있습니다.

16.8. KICKSTART 부팅 옵션

이 섹션에서는 **Kickstart** 파일에 추가하여 설치를 자동화할 수 있는 부팅 옵션에 대해 설명합니다.

inst.ks=

설치를 자동화하는 데 사용할 **Kickstart** 파일의 위치를 정의합니다. **inst.repo** 형식을 사용하여 위치를 지정할 수 있습니다. 장치가 아니라 경로를 지정하면 설치 프로그램은 지정된 장치에서 **/ks.cfg**에서 **Kickstart** 파일을 찾습니다.

장치를 지정하지 않고 이 옵션을 사용하는 경우 설치 프로그램은 옵션에 다음 값을 사용합니다.

```
inst.ks=nfs:next-server:/filename
```

이전 예에서 **next-server**는 **DHCP next-server** 옵션 또는 **DHCP** 서버 자체의 **IP** 주소이며 **filename**은 **DHCP** 파일 이름 옵션 또는 **/kickstart/**입니다. 지정된 파일 이름이 **/** 문자로 종료되면 **ip-kickstart**가 추가됩니다. 다음 표에는 예제가 포함되어 있습니다.

표 **16.5.** 기본 **Kickstart** 파일 위치

DHCP 서버 주소	클라이언트 주소	Kickstart 파일 위치
192.168.122.1	192.168.122.100	192.168.122.1:/kickstart/192.168.122.100-kickstart

OEMDRV 레이블이 있는 볼륨이 있는 경우 설치 프로그램은 **ks.cfg** 라는 **Kickstart** 파일을 로드하려고 합니다. **Kickstart** 파일이 이 위치에 있는 경우 **inst.ks=** 부팅 옵션을 사용할 필요가 없습니다.

inst.ks.all

inst.ks.all 옵션을 지정하여 여러 **inst.ks** 옵션에서 제공하는 여러 **Kickstart** 파일 위치를 순차적으로 시도합니다. 첫 번째 위치가 사용됩니다. 이는 유형 **http,https** 또는 **ftp**의 위치에만 적용되며 다른 위치는 무시됩니다.

inst.ks.sendmac

inst.ks.sendmac 옵션을 사용하여 모든 네트워크 인터페이스의 **MAC** 주소가 포함된 발신 **HTTP** 요청에 헤더를 추가합니다. 예를 들어 다음과 같습니다.

```
X-RHN-Provisioning-MAC-0: eth0 01:23:45:67:89:ab
```

이 기능은 **inst.ks=http**를 사용하여 시스템을 프로비저닝할 때 유용할 수 있습니다.

inst.ks.sendsn

inst.ks.sendsn 옵션을 사용하여 발신 **HTTP** 요청에 헤더를 추가합니다. 이 헤더에는 **/sys/class/dmi/id/product_serial**에서 읽은 시스템 일련 번호가 포함되어 있습니다. 헤더에는 다음 구문이 있습니다.

```
X-System-Serial-Number: R8VA23D
```

추가 리소스

- 전체 부팅 옵션 목록

16.9. 고급 설치 부팅 옵션

이 섹션에는 고급 설치 부팅 옵션에 대한 정보가 포함되어 있습니다.

inst.kexec

inst.kexec 옵션은 새 시스템을 즉시 로드하고 **BIOS** 또는 펌웨어에서 일반적으로 수행하는 하드웨어 초기화를 바이패스합니다.

중요

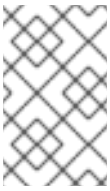
이 옵션은 더 이상 사용되지 않으며 기술 프리뷰로만 제공됩니다. 기술 프리뷰 기능에 대한 **Red Hat** 지원 범위 정보는 기술 프리뷰 기능 지원 범위 문서를 참조하십시오.

inst.multilib

*이는 **%packages** 섹션에 직접 지정된 패키지에만 적용됩니다. 패키지가 종속성으로 설치된 경우 정확히 지정된 종속성만 설치됩니다.*

selinux=0

*기본적으로 **SELinux**는 설치 프로그램에서 허용 모드로 작동하며 설치된 시스템의 강제 모드에서 작동합니다.*



참고

inst.nonibftiscsiboot

16.10.

method

dns

netmask, gateway, hostname

ip=bootif

ksdevice

표 16.6.

값	정보
	해당 없음
ksdevice=link	
ksdevice=bootif	
ksdevice=ibft	
ksdevice=<MAC>	
ksdevice=<DEV>	

16.11. 제거된 부팅 옵션

*이 섹션에는 **Red Hat Enterprise Linux**에서 제거된 부팅 옵션이 포함되어 있습니다.*

참고

askmethod, asknetwork

blacklist, nofirewire

inst.headless=

inst.decorated

repo=nfsiso

ethtool

gdb

inst.mediacheck

ks=floppy

utf8

noipv6

ipv6는 커널에 빌드되며 설치 프로그램에서 제거할 수 없습니다.

upgradeany

17장. UEFI SECURE BOOT를 사용하여 베타 시스템 부팅

운영 체제의 보안을 강화하려면 UEFI Secure Boot가 활성화된 시스템에서 Red Hat Enterprise Linux 베타 릴리스를 부팅할 때 UEFI Secure Boot 기능을 사용하여 서명 확인을 수행합니다.

17.1. UEFI SECURE BOOT 및 RHEL BETA 릴리스

UEFI Secure Boot를 사용하려면 운영 체제 커널이 인식된 개인 키로 서명해야 합니다. UEFI Secure Boot는 해당 공개 키를 사용하여 서명을 확인합니다.

Red Hat Enterprise Linux 베타 릴리스의 경우 커널은 Red Hat 베타 고유의 개인 키로 서명됩니다. UEFI Secure Boot는 해당 공개 키를 사용하여 서명을 확인하려고 하지만 하드웨어가 베타 개인 키를 인식하지 않기 때문에 Red Hat Enterprise Linux Beta 릴리스 시스템이 부팅되지 않습니다. 따라서 베타 릴리스와 함께 UEFI Secure Boot를 사용하려면 MOK(Machine Owner Key) 기능을 사용하여 시스템에 Red Hat 베타 공개 키를 추가합니다.

17.2. UEFI SECURE BOOT의 베타 공개 키 추가

이 섹션에서는 UEFI Secure Boot를 위한 Red Hat Enterprise Linux 베타 공개 키를 추가하는 방법에 대해 설명합니다.

사전 요구 사항

- 시스템에서 UEFI Secure Boot가 비활성화되어 있습니다.
- Red Hat Enterprise Linux 베타 릴리스가 설치되고 시스템 재부팅 후에도 Secure Boot가 비활성화됩니다.
- 시스템에 로그인한 후 Initial Setup 창의 작업이 완료됩니다.

절차

1. 시스템의 MOK(Machine Owner Key) 목록에 Red Hat 베타 공개 키 등록을 시작합니다.

```
# mokutil --import /usr/share/doc/kernel-keys/$(uname -r)/kernel-signing-ca.cer
```

$(uname -r)은 커널 버전 (예: 4.18.0-80.el8.x86_64)으로 교체됩니다.

2. *메시지가 표시되면 암호를 입력합니다.*

3. *시스템을 재부팅하고 임의의 키를 눌러 시작합니다. Shim UEFI 키 관리 유틸리티는 시스템을 시작하는 동안 시작됩니다.*

4. *Enroll MOK를 선택합니다.*

5. *Continue를 선택합니다.*

6. *Yes를 선택하고 암호를 입력합니다. 키를 시스템의 펌웨어로 가져옵니다.*

7. *Reboot를 선택합니다.*

8. *시스템에서 Secure Boot를 활성화합니다.*

17.3. 베타 공개 키 제거

Red Hat Enterprise Linux 베타 릴리스를 제거하고 Red Hat Enterprise Linux General Availability (GA) 릴리스 또는 다른 운영 체제를 설치하려는 경우 베타 공개 키를 제거하십시오.

이 절차에서는 베타 공개 키를 제거하는 방법을 설명합니다.

절차

1. *시스템의 MOK(Machine Owner Key) 목록에서 Red Hat 베타 공개 키를 제거하십시오.*

```
# mokutil --reset
```

2. *메시지가 표시되면 암호를 입력합니다.*

3. *시스템을 재부팅하고 임의의 키를 눌러 시작합니다. **Shim UEFI** 키 관리 유틸리티는 시스템을 시작하는 동안 시작됩니다.*

4. ***Reset MOK**을 선택합니다.*

5. ***Continue**를 선택합니다.*

6. ***Yes**를 선택하고 **2**단계에서 지정한 암호를 입력합니다. 키는 시스템 펌웨어에서 제거됩니다.*

7. ***Reboot**를 선택합니다.*

V 부. KICKSTART 참조

부록 A. KICKSTART 스크립트 파일 형식 참조

이 참조는 Kickstart 파일 형식을 자세히 설명합니다.

A.1. KICKSTART 파일 형식

Kickstart 스크립트는 설치 프로그램에서 인식하는 키워드가 포함된 일반 텍스트 파일입니다. 이 파일은 설치에 대한 지침으로 사용됩니다. Linux 시스템에서 Gedit이나 vim 또는 Windows 시스템의 Notepad와 같은 ASCII 텍스트로 파일을 저장할 수 있는 텍스트 편집기를 사용하여 Kickstart 파일을 만들고 편집할 수 있습니다. Kickstart 구성의 파일 이름은 중요하지 않지만 나중에 다른 구성 파일 또는 대화 상자에서 이 이름을 지정해야 하므로 간단한 이름을 사용하는 것이 좋습니다.

명령

명령은 설치에 대한 지침으로 사용되는 키워드입니다. 각 명령은 한 줄에 있어야 합니다. 명령은 옵션을 사용할 수 있습니다. 명령 및 옵션 지정은 쉘에서 Linux 명령을 사용하는 것과 유사합니다.

섹션

퍼센트 %로 시작하는 특정 특수 명령은 섹션을 시작합니다. 섹션의 명령 해석은 섹션 외부에 배치된 명령과 다릅니다. 모든 섹션은 %end 명령으로 완료되어야 합니다.

섹션 유형

사용 가능한 섹션은 다음과 같습니다.

- 애드온 섹션. 이러한 섹션에서는 %addon addon_name 명령을 사용합니다.
- 패키지 선택 섹션. %packages로 시작합니다. 이 파일을 사용하여 패키지 그룹 또는 모듈과 같은 간접 수단을 포함하여 설치용 패키지를 나열합니다.
- 스크립트 섹션. 이러한 작업은 %pre, %pre-install, %post, %onerror로 시작합니다. 이러한 섹션은 필수가 아닙니다.

명령 섹션

명령 섹션은 스크립트 섹션 또는 %packages 섹션에 포함되지 않은 Kickstart 파일의 명령에 사용되는 용어입니다.

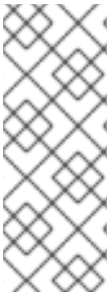
스크립트 섹션 수 및 순서

명령 섹션을 제외한 모든 섹션은 선택 사항이며 여러 번 존재할 수 있습니다. 특정 유형의 스크립트 섹션을 평가할 때 **Kickstart**에 존재하는 유형의 모든 섹션이 표시되는 순서대로 두 **%post** 섹션이 차례로 평가됩니다. 그러나 다양한 유형의 스크립트 섹션을 순서대로 지정할 필요는 없습니다. **%pre** 섹션 앞에 **%post** 섹션이 있는지는 중요하지 않습니다.

주석

Kickstart 주석은 해시 **#** 문자로 시작하는 행입니다. 이러한 행은 설치 프로그램에서 무시됩니다.

필요하지 않은 항목은 생략할 수 있습니다. 필요한 항목을 생략하면 설치 프로그램이 대화식 모드로 변경되어 사용자가 일반 대화식 설치와 마찬가지로 사용자가 관련 항목에 대한 답변을 제공할 수 있습니다. 또한 **cmdline** 명령을 사용하여 **kickstart** 스크립트를 비대화형으로 선언할 수도 있습니다. 비대화형 모드에서는 답변이 누락되면 설치 프로세스가 중단됩니다.



참고

텍스트 또는 그래픽 모드로 **Kickstart**를 설치하는 동안 사용자 상호 작용이 필요한 경우 설치를 완료하기 위해 업데이트가 필요한 창만 입력합니다. **spokes**를 입력하면 **Kickstart** 구성이 재설정될 수 있습니다. 설정 재설정은 특히 설치 대상 창을 입력한 후 스토리지와 관련된 **Kickstart** 명령에 적용됩니다.

A.2. KICKSTART의 패키지 선택

Kickstart는 설치할 패키지를 선택하기 위해 **%packages** 명령으로 시작된 섹션을 사용합니다. 이러한 방식으로 패키지, 그룹, 환경, 모듈 스트림 및 모듈 프로필을 설치할 수 있습니다.

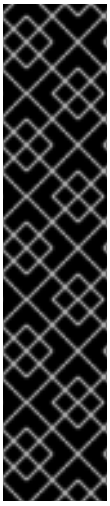
A.2.1. 패키지 선택 섹션

설치할 소프트웨어 패키지를 설명하는 **Kickstart** 섹션을 시작하려면 **%packages** 명령을 사용합니다. **%packages** 섹션은 **%end** 명령으로 끝나야 합니다.

환경, 그룹, 모듈 스트림, 모듈 프로필 또는 해당 패키지 이름으로 패키지를 지정할 수 있습니다. 관련 패키지가 포함된 여러 환경 및 그룹이 정의됩니다.

각 항목에는 **ID**, 사용자 가시성 값, 이름, 설명 및 패키지 목록이 있습니다.

이로 인해 시스템이 취약점의 영향을 받을 가능성이 크게 줄어듭니다. 필요한 경우 설치 후 나중에 추가 패키지를 추가할 수 있습니다. 데스크탑 환경과 **X Window** 시스템이 설치에 포함되어 있고 그래픽 로그인이 활성화된 경우가 아니면 **Kickstart** 파일에서 시스템을 설치한 후에는 초기 설정을 실행할 수 없습니다.



중요

64비트 시스템에 **32**비트 패키지를 설치하려면 다음을 수행합니다.

- **%packages** 섹션에 **--multilib** 옵션을 지정합니다.
- 패키지가 빌드된 **32**비트 아키텍처(예: **glibc.i686**)를 사용하여 패키지 이름을 추가합니다.

A.2.2. 패키지 선택 명령

이러한 명령은 **Kickstart** 파일의 **%packages** 섹션에서 사용할 수 있습니다.

환경 지정

@^ 기호로 시작하는 줄로 설치할 전체 환경을 지정합니다.

```
%packages
@^Infrastructure Server
%end
```

그러면 **Infrastructure Server** 환경의 일부인 모든 패키지가 설치됩니다.

Kickstart 파일에 단일 환경만 지정해야 합니다. 더 많은 환경이 지정되면 마지막으로 지정된 환경만 사용됩니다.

그룹 지정

예를 들어 다음과 같습니다.

```
%packages
@X Window System
@Desktop
```

```
@Sound and Video
%end
```

Core 그룹은 항상 선택됩니다. **%packages** 섹션에 지정할 필요는 없습니다.

개별 패키지 지정

개별 패키지를 이름으로 한 줄에 하나의 항목을 지정합니다. 별표 문자(*)를 패키지 이름에서 와일드카드로 사용할 수 있습니다. 예를 들어 다음과 같습니다.

```
%packages
sqlite
curl
aspell
docbook*
%end
```

모듈 스트림의 프로필 지정

프로필 구문을 사용하여 한 줄에 한 항목씩 모듈 스트림에 대한 프로필을 지정합니다.

```
%packages
@module:stream/profile
%end
```

이렇게 하면 모듈 스트림의 지정된 프로필에 나열된 모든 패키지가 설치됩니다.

- 모듈에 기본 스트림을 지정하면 해당 스트림을 해제할 수 있습니다. 기본 스트림을 지정하지 않으면 이를 지정해야 합니다.
- 모듈 스트림에 기본 프로필이 지정되면 그대로 둘 수 있습니다. 기본 프로필을 지정하지 않으면 이 프로필을 지정해야 합니다.
- 다른 스트림을 사용하여 모듈을 여러 번 설치할 수 없습니다.
- 동일한 모듈과 스트림의 여러 프로필을 설치할 수 있습니다.

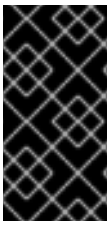
모듈과 그룹은 @ 기호로 시작하는 것과 동일한 구문을 사용합니다. 동일한 이름의 모듈 및 패키지 그룹이 있는 경우 모듈이 우선합니다.

또한 **module Kickstart** 명령을 사용하여 모듈 스트림을 활성화한 다음 직접 이름을 지정하여 모듈 스트림에 포함된 패키지를 설치할 수도 있습니다.

환경, 그룹 또는 패키지 제외

선행 대시(**-**)를 사용하여 설치에서 제외할 패키지 또는 그룹을 지정합니다. 예를 들어 다음과 같습니다.

```
%packages
-@Graphical Administration Tools
-autofs
-ipa*compat
%end
```



중요

Kickstart 파일에서 * 만 사용하여 사용 가능한 모든 패키지를 설치하는 것은 지원되지 않습니다.

여러 옵션을 사용하여 **%packages** 섹션의 기본 동작을 변경할 수 있습니다. 일부 옵션은 전체 패키지 선택에서 작동하며 다른 옵션은 특정 그룹에만 사용됩니다.

추가 리소스

- 소프트웨어 설치
- 사용자 공간 구성 요소 설치, 관리 및 제거

A.2.3. 일반적인 패키지 선택 옵션

%packages 섹션에 다음 옵션을 사용할 수 있습니다. 옵션을 사용하려면 패키지 선택 섹션의 시작 부분에 추가합니다. 예를 들어 다음과 같습니다.

```
%packages --multilib --ignoremissing
```

--default

기본 패키지 세트를 설치합니다. 이는 대화형 설치 중에 **Package Selection** 화면에서 다른 선택 항목이 없는 경우 설치되는 패키지 세트에 해당합니다.

--excludedocs

패키지에 포함된 문서는 설치하지 마십시오.

--ignoremissing

--instLangs=

설치할 언어 목록을 지정합니다. 이 값은 패키지 그룹 수준 선택과 다릅니다.

--multilib

multilib 패키지에 설치된 시스템을 구성하고, **64**비트 시스템에 **32**비트 패키지를 설치할 수 있도록 구성하고, 이와 같이 이 섹션에 지정된 패키지를 설치합니다.

일반적으로 **AMD64** 및 **Intel 64** 시스템에서는 **x86_64** 및 **noarch** 패키지만 설치할 수 있습니다. 그러나 **multilib** 옵션을 사용하면 **32**비트 **AMD** 및 **i686 Intel** 시스템 패키지가 있는 경우 자동으로 설치할 수 있습니다.

이는 **%packages** 섹션에 명시적으로 지정된 패키지에만 적용됩니다. **Kickstart** 파일에 지정되지 않고 종속 항목으로만 설치되는 패키지는 더 많은 아키텍처에서 사용할 수 있더라도 필요한 아키텍처 버전에만 설치됩니다.

사용자는 시스템을 설치하는 동안 **multilib** 모드에서 패키지를 설치하도록 **Anaconda**를 구성할 수 있습니다. 다음 옵션 중 하나를 사용하여 **multilib** 모드를 활성화합니다.

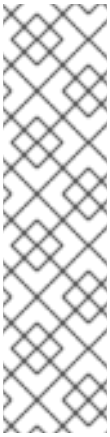
1. 다음 행을 사용하여 **Kickstart** 파일을 설정합니다.

```
%packages --multilib --default
%end
```

2. *설치 이미지를 부팅하는 동안 **inst.multilib** 부팅 옵션을 추가합니다.*

--nocore

*그렇지 않으면 기본적으로 설치된 **@Core** 패키지 그룹의 설치를 비활성화합니다. **--nocore** 로 **@Core** 패키지 그룹을 비활성화하면 경량 컨테이너를 생성하는 데만 사용해야 합니다. **--nocore**를 사용하여 데스크탑 또는 서버 시스템을 설치하면 시스템을 사용할 수 없게 됩니다.*



참고

- ***-@Core**를 사용하여 **@Core** 패키지 그룹에서 패키지를 제외하면 작동하지 않습니다. **@Core** 패키지 그룹을 제외하는 유일한 방법은 **--nocore** 옵션을 사용하는 것입니다.*
- ***@Core** 패키지 그룹은 작동 중인 시스템을 설치하는 데 필요한 최소한의 패키지 세트로 정의됩니다.*

--excludeWeakdeps

약한 종속성에서 패키지 설치를 비활성화합니다.

--retries=

*기본값은 **10**입니다.*

--timeout=

*기본값은 **30**입니다.*

A.2.4. 특정 패키지 그룹 옵션

*이 목록의 옵션은 단일 패키지 그룹에만 적용됩니다. **Kickstart** 파일의 **%packages** 명령에 사용하는 대신 그룹 이름에 추가합니다. 예를 들어 다음과 같습니다.*

```
%packages
@Graphical Administration Tools --optional
%end
```

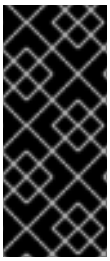
--nodefaults

기본 선택 항목이 아닌 그룹의 필수 패키지만 설치합니다.

--optional

기본 선택 항목을 설치하는 것 외에도 ***-comps-repository.architecture.xml** 파일의 그룹 정의에서 선택 사항으로 표시된 패키지를 설치합니다.

Scientific Support와 같은 일부 패키지 그룹에는 필수 또는 기본 패키지가 지정되지 않음(선택 사항)이 없습니다. 이 경우 **--optional** 옵션을 항상 사용해야 합니다. 그렇지 않으면 이 그룹의 패키지가 설치되지 않습니다.



중요

--nodefaults 및 **--optional** 옵션은 함께 사용할 수 없습니다. **--nodefaults**를 사용하여 설치 중에 필수 패키지만 설치하고 설치된 시스템 사후 설치에 선택적 패키지를 설치할 수 있습니다.

A.3. KICKSTART 파일의 스크립트

Kickstart 파일은 다음 스크립트를 포함할 수 있습니다.

- **%pre**
- **%pre-install**
- **%post**

이 섹션에서는 스크립트에 대한 다음 세부 정보를 제공합니다.

- 실행 시간
- 스크립트에 포함될 수 있는 명령 유형입니다.

- 스크립트의 목적
- 스크립트 옵션

A.3.1. %pre 스크립트

%pre 스크립트는 **Kickstart** 파일이 로드된 직후 시스템에서 실행되지만 완전히 구문 분석되고 설치가 시작되기 전에 실행됩니다. 이러한 각 섹션은 **%pre**로 시작하고 **%end**로 끝나야 합니다.

%pre 스크립트는 네트워킹 및 스토리지 장치의 활성화 및 구성에 사용할 수 있습니다. 설치 환경에서 사용 가능한 인터프리터를 사용하여 스크립트를 실행할 수도 있습니다.

%pre 스크립트의 문제를 디버깅하기 어려울 수 있으므로 필요한 경우에만 **%pre** 스크립트를 사용하는 것이 좋습니다.

중요

네트워킹, 스토리지 및 파일 시스템과 관련된 명령은 설치 환경 **/sbin** 및 **/bin** 디렉터리에 있는 대부분의 유틸리티 외에 **%pre** 스크립트에서 사용할 수 있습니다.

%pre 섹션에서 네트워크에 액세스할 수 있습니다. 그러나 이 시점에서 이름 서비스가 구성되지 않았으므로 **URL**이 아닌 **IP** 주소만 작동합니다.

참고

pre 스크립트는 **chroot** 환경에서 실행되지 않습니다.

A.3.1.1. %pre 스크립트 섹션 옵션

다음 옵션을 사용하여 사전 설치 스크립트의 동작을 변경할 수 있습니다. 옵션을 사용하려면 스크립트 시작 시 **%pre** 줄에 추가합니다. 예를 들어 다음과 같습니다.

```
%pre --interpreter=/usr/libexec/platform-python
-- Python script omitted --
%end
```

--interpreter=

Python과 같은 다른 스크립팅 언어를 지정할 수 있습니다. 시스템에서 사용 가능한 모든 스크립팅 언어를 사용할 수 있습니다. 대부분의 경우 **/usr/bin/sh**,**/usr/bin/bash** 및 **/usr/libexec/platform-python** 입니다.

platform-python 인터프리터는 **Python** 버전 **3.6**을 사용합니다. 새 경로와 버전의 **Python** 스크립트를 이전 **RHEL** 버전에서 변경해야 합니다.

--erroronfail

스크립트가 실패하면 오류를 표시하고 설치를 중지합니다. 오류 메시지는 실패의 원인이 기록되는 위치로 안내합니다. 설치된 시스템은 불안정하고 부팅 불가능한 상태가 될 수 있습니다. **inst.nokill** 옵션을 사용하여 스크립트를 디버깅할 수 있습니다.

--log=

스크립트의 출력을 지정된 로그 파일에 기록합니다. 예를 들어 다음과 같습니다.

```
%pre --log=/tmp/ks-pre.log
```

A.3.2. %pre-install 스크립트

pre-install 스크립트의 명령은 다음 작업이 완료된 후 실행됩니다.

- 시스템이 분할됨
- 파일 시스템은 **/mnt/sysroot**에 생성 및 마운트됨
- 네트워크가 부팅 옵션 및 **Kickstart** 명령에 따라 구성되었습니다.

%pre-install 각 섹션은 **%pre-install**로 시작하고 **%end**로 끝나야 합니다.

%pre-install 스크립트를 사용하여 설치를 수정하고 패키지 설치 전에 보장된 **ID**가 있는 사용자 및 그

룹을 추가할 수 있습니다.

*설치에 필요한 수정 사항에 대해 **%post** 스크립트를 사용하는 것이 좋습니다. **%post** 스크립트가 필요한 수정 사항에 대한 짧은 경우에만 **%pre-install** 스크립트를 사용합니다.*

A.3.2.1. %pre-install 스크립트 섹션 옵션

*다음 옵션을 사용하여 **pre-install** 스크립트의 동작을 변경할 수 있습니다. 옵션을 사용하려면 스크립트 시작 시 **%pre-install** 행에 추가합니다. 예를 들어 다음과 같습니다.*

```
%pre-install --interpreter=/usr/libexec/platform-python
-- Python script omitted --
%end
```

*동일한 인터프리터와 함께 **%pre-install** 섹션이 여러 개 있을 수 있습니다. **Kickstart** 파일에 나타나는 순서에 따라 평가됩니다.*

--interpreter=

***Python**과 같은 다른 스크립팅 언어를 지정할 수 있습니다. 시스템에서 사용 가능한 모든 스크립팅 언어를 사용할 수 있습니다. 대부분의 경우 **/usr/bin/sh**,**/usr/bin/bash** 및 **/usr/libexec/platform-python** 입니다.*

***platform-python** 인터프리터는 **Python** 버전 **3.6**을 사용합니다. 새 경로와 버전의 **Python** 스크립트를 이전 **RHEL** 버전에서 변경해야 합니다.*

--erroronfail

*스크립트가 실패하면 오류를 표시하고 설치를 중지합니다. 오류 메시지는 실패의 원인이 기록되는 위치로 안내합니다. 설치된 시스템은 불안정하고 부팅 불가능한 상태가 될 수 있습니다. **inst.nokill** 옵션을 사용하여 스크립트를 디버깅할 수 있습니다.*

--log=

스크립트의 출력을 지정된 로그 파일에 기록합니다. 예를 들어 다음과 같습니다.

```
%pre-install --log=/mnt/sysroot/root/ks-pre.log
```

A.3.3. %post 스크립트

%post 스크립트는 설치가 완료된 후에 실행되는 설치 후 스크립트이지만 시스템을 처음 재부팅하기 전에 실행됩니다. 이 섹션을 사용하여 시스템 서브스크립션과 같은 작업을 실행할 수 있습니다.

설치가 완료되면 시스템에서 실행할 명령을 추가하는 옵션이 있지만 시스템을 처음 재부팅하기 전에 실행할 수 있습니다. 이 섹션은 **%post** 로 시작하고 **%end** 로 끝나야 합니다.

%post 섹션은 추가 소프트웨어 설치 또는 추가 이름 서버 구성과 같은 기능에 유용합니다. 설치 후 스크립트는 **chroot** 환경에서 실행되므로 설치 미디어에서 스크립트 또는 **RPM** 패키지를 복사하는 것은 기본적으로 작동하지 않습니다. 아래 설명된 대로 **--nochroot** 옵션을 사용하여 이 동작을 변경할 수 있습니다. 그런 다음 **%post** 스크립트가 설치된 대상 시스템의 **chroot** 가 아닌 설치 환경에서 실행됩니다.

설치 후 스크립트는 **chroot** 환경에서 실행되므로 대부분의 **systemctl** 명령은 모든 작업 수행을 거부합니다.

%post 섹션을 실행하는 동안 설치 미디어가 여전히 삽입되어야 합니다.

A.3.3.1. %post 스크립트 섹션 옵션

다음 옵션을 사용하여 설치 후 스크립트의 동작을 변경할 수 있습니다. 옵션을 사용하려면 스크립트 시작 부분에 있는 **%post** 줄에 추가합니다. 예를 들어 다음과 같습니다.

```
%post --interpreter=/usr/libexec/platform-python
-- Python script omitted --
%end
```

--interpreter=

Python과 같은 다른 스크립팅 언어를 지정할 수 있습니다. 예를 들어 다음과 같습니다.

```
%post --interpreter=/usr/libexec/platform-python
```

시스템에서 사용 가능한 모든 스크립팅 언어를 사용할 수 있습니다. 대부분의 경우 **/usr/bin/sh**,**/usr/bin/bash** 및 **/usr/libexec/platform-python** 입니다.

platform-python 인터프리터는 **Python** 버전 **3.6**을 사용합니다. 새 경로와 버전의 **Python** 스크립트를 이전 **RHEL** 버전에서 변경해야 합니다.

--nochroot

chroot 환경 외부에서 실행하려는 명령을 지정할 수 있습니다.

다음 예제에서는 **/etc/resolv.conf** 파일을 방금 설치한 파일 시스템에 복사합니다.

```
%post --nochroot
cp /etc/resolv.conf /mnt/sysroot/etc/resolv.conf
%end
```

--erroronfail

스크립트가 실패하면 오류를 표시하고 설치를 중지합니다. 오류 메시지는 실패의 원인이 기록되는 위치로 안내합니다. 설치된 시스템은 불안정하고 부팅 불가능한 상태가 될 수 있습니다. **inst.nokill** 옵션을 사용하여 스크립트를 디버깅할 수 있습니다.

--log=

스크립트의 출력을 지정된 로그 파일에 기록합니다. 로그 파일의 경로는 **--nochroot** 옵션을 사용할지 여부를 고려해야 합니다. 예를 들어 **--nochroot** 가 없는 경우:

```
%post --log=/root/ks-post.log
```

--nochroot 와 함께 다음을 수행합니다.

```
%post --nochroot --log=/mnt/sysroot/root/ks-post.log
```

A.3.3.2.

%post 섹션의 예에서는 **NFS** 공유를 마운트하고 공유의 **/usr/new-machines/** 에 있는 **runme** 라는 스크립트를 실행합니다. **Kickstart** 모드에서는 **NFS** 파일 잠금이 지원되지 않으므로 **-o nolock** 옵션이 필요합니다.

```
# Start of the %post section with logging into /root/ks-post.log
%post --log=/root/ks-post.log

# Mount an NFS share
mkdir /mnt/temp
mount -o nolock 10.10.0.2:/usr/new-machines /mnt/temp
openvt -s -w -- /mnt/temp/runme
umount /mnt/temp

# End of the %post section
%end
```

A.3.3.3.

```
%post --log=/root/ks-post.log
subscription-manager register --username=admin@example.com --password=secret --auto-attach
%end
```

A.4.

예 A.1.

```
%anaconda
pwpolicy root --minlen=10 --strict
%end
```

A.5. KICKSTART 오류 처리 섹션

Red Hat Enterprise Linux 7부터 Kickstart 설치에는 설치 프로그램이 치명적인 오류가 발생할 때 실행되는 사용자 지정 스크립트가 포함될 수 있습니다. 이러한 오류가 발생한 후에는 설치를 계속할 수 없습니다. 설치 프로그램은 Kickstart 파일에서 제공되는 모든 %onerror 스크립트를 실행합니다.

%end로 종료하려면 각 %onerror 스크립트가 필요합니다.

오류 처리 섹션에는 다음 옵션을 사용할 수 있습니다.

--erroronfail

스크립트가 실패하면 오류를 표시하고 설치를 중지합니다. 오류 메시지는 실패의 원인이 기록되는 위치로 안내합니다. 설치된 시스템은 불안정하고 부팅 불가능한 상태가 될 수 있습니다. inst.nokill 옵션을 사용하여 스크립트를 디버깅할 수 있습니다.

--interpreter=

Python과 같은 다른 스크립팅 언어를 지정할 수 있습니다. 예를 들어 다음과 같습니다.

```
%onerror --interpreter=/usr/libexec/platform-python
```

시스템에서 사용 가능한 모든 스크립팅 언어를 사용할 수 있습니다. 대부분의 경우 /usr/bin/sh,/usr/bin/bash 및 /usr/libexec/platform-python 입니다.

platform-python 인터프리터는 Python 버전 3.6을 사용합니다. 새 경로와 버전의 Python 스크립트를 이전 RHEL 버전에서 변경해야 합니다.

--log=

스크립트의 출력을 지정된 로그 파일에 기록합니다.

A.6. KICKSTART 애드온 섹션

이러한 추가 기능은 다양한 방식으로 기본 Kickstart(및 Anaconda) 기능을 확장할 수 있습니다.

Kickstart 파일에서 애드온을 사용하려면 %addon addon_name options 명령을 사용하고 사전 설치 및 설치 후 스크립트 섹션과 유사하게 %end 문과 함께 명령을 완료합니다. 예를 들어 기본적으로 Anaconda와 함께 배포되는 Kdump 애드온을 사용하려면 다음 명령을 사용합니다.

```
%addon com_redhat_kdump --enable --reserve-mb=auto
%end
```

***%addon** 명령에는 자체적으로 옵션이 포함되어 있지 않습니다. 모든 옵션은 실제 애드온에 따라 다릅니다.*

부록 B. KICKSTART 명령 및 옵션 참조

이 참조는 Red Hat Enterprise Linux 설치 프로그램 프로그램에서 지원하는 모든 Kickstart 명령의 전체 목록입니다. 명령은 몇 가지 광범위한 카테고리로 알파벳순으로 정렬됩니다. 명령이 여러 카테고리에 속할 수 있는 경우 모든 카테고리에 나열됩니다.

B.1. KICKSTART 변경

RHEL 8에서 auth 또는 authconfig가 더 이상 사용되지 않음

이 호환성 계층 및 알려진 문제에 대한 설명은 수동 페이지 authselect-migration(7)을 참조하십시오. 설치 프로그램은 더 이상 사용되지 않는 명령의 사용을 자동으로 감지하고 호환성 계층을 제공하기 위해 authselect-compat 패키지를 시스템에 설치합니다.

Kickstart는 더 이상 Btrfs를 지원하지 않음

Red Hat Enterprise Linux 8에서는 RuntimeClass 파일 시스템이 지원되지 않습니다. 그 결과 Graphical User Interface(GPU)와 Kickstart 명령이 더 이상 vGPU를 지원하지 않습니다.

이전 RHEL 릴리스의 Kickstart 파일 사용

B.1.1. 더 이상 사용되지 않는 Kickstart 명령 및 옵션

특정 옵션만 나열된 경우에도 기본 명령 및 기타 옵션은 계속 사용할 수 있으며 더 이상 사용되지 않습니다.

-

- *device*
- *deviceprobe*
- *dmraid*
- *install - 하위 명령 또는 방법을 명령으로 직접 사용*
- *multipath*
- *bootloader --upgrade*
- *ignoredisk --interactive*
- *partition --active*
- *reboot --kexec*

B.1.2. 제거된 Kickstart 명령 및 옵션

Kickstart 파일에서 사용하면 오류가 발생합니다.

- *device*
- *deviceprobe*

- ***dmraid***
- ***install - 하위 명령 또는 방법을 명령으로 직접 사용***
- ***multipath***
- ***bootloader --upgrade***
- ***ignoredisk --interactive***
- ***partition --active***
- ***harddrive --biospart***
- ***업그레이드 (이 명령은 이전에 더 이상 사용되지 않았습니다.)***
- ***btrfs***
- ***part/partition btrfs***
- ***part --fstype btrfs 또는 partition --fstype btrfs***
- ***logvol --fstype btrfs***
- ***RAID --fstype btrfs***
- ***unsupported_hardware***

특정 옵션과 값만 나열된 경우 기본 명령 및 기타 옵션을 계속 사용할 수 있으며 제거되지 않습니다.

B.2. 설치 프로그램 구성 및 흐름 제어를 위한 KICKSTART 명령

*이 목록의 **Kickstart** 명령은 설치 모드와 설치 과정을 제어하며 결국 수행되는 작업을 제어합니다.*

B.2.1. cdrom

***cdrom Kickstart** 명령은 선택 사항입니다. 시스템의 첫 번째 광 드라이브의 설치를 수행합니다.*

구문

```
cdrom
```

참고

- *이전에는 **cdrom** 명령을 **install** 명령과 함께 사용해야 했습니다. **install** 명령이 더 이상 사용되지 않으며 **cdrom** 은 설치를 의미하기 때문에 자체적으로 사용할 수 있습니다.*
- *이 명령에는 옵션이 없습니다.*
- *실제로 설치를 실행하려면 **cdrom,hard drive,hmc,nfs,liveimg** 또는 **url** 중 하나를 지정해야 합니다.*

B.2.2. cmdline

***cmdline Kickstart** 명령은 선택 사항입니다. 완전히 비대화형 명령줄 모드에서 설치를 수행합니다. 상호 작용에 대한 프롬프트가 표시되면 설치가 중지됩니다.*

구문

```
cmdline
```

참고

- 완전히 자동 설치의 경우 사용 가능한 모드(**graphical**,**text** 또는 **cmdline**) 중 하나를 **Kickstart** 파일에서 지정해야 합니다. 그렇지 않으면 **console=** 부팅 옵션을 사용해야 합니다. 모드가 지정되지 않은 경우 시스템은 그래픽 모드를 사용하거나 **VNC** 및 텍스트 모드에서 선택하라는 메시지를 표시합니다.
- 이 명령에는 옵션이 없습니다.
- 이 모드는 **x3270** 터미널을 사용하는 **64**비트 **IBM Z** 시스템에서 유용합니다.

B.2.3. driverdisk

driverdisk Kickstart 명령은 선택 사항입니다. 이를 사용하여 설치 프로그램에 추가 드라이버를 제공합니다.

Kickstart 설치 중에 드라이버 디스크를 사용하여 기본적으로 포함되지 않은 추가 드라이버를 제공할 수 있습니다. 드라이버 디스크 콘텐츠를 시스템의 하드 드라이브에 있는 파티션의 루트 디렉터리에 복사해야 합니다. 그런 다음 **driverdisk** 명령을 사용하여 설치 프로그램이 드라이버 디스크와 해당 위치를 찾도록 지정해야 합니다.

구문

```
driverdisk [partition|--source=url|--biospart=biospart]
```

옵션

다음 중 한 가지 방법으로 드라이버 디스크의 위치를 지정해야 합니다.

- **partition -** 드라이버 디스크를 포함하는 파티션입니다. 파티션을 파티션 이름(예: **sdb1**)뿐

만 아니라 전체 경로(예: /dev/ sdb1)로 지정해야 합니다.

- --source= - 드라이버 디스크의 URL입니다. 예를 들면 다음과 같습니다.

```
driverdisk --source=ftp://path/to/dd.img
driverdisk --source=http://path/to/dd.img
driverdisk --source=nfs:host:/path/to/dd.img
```

- --biospart= - 드라이버 디스크가 포함된 BIOS 파티션(예: 82p2)

참고

드라이버 디스크는 또한 하드 디스크 드라이브 또는 네트워크를 통해 로드되는 대신 유사한 장치에서 또는 initrd에서 로드할 수 있습니다. 다음 절차를 따르십시오.

1. 하드 디스크 드라이브, USB 또는 유사한 장치에 드라이버 디스크를 로드합니다.
2. 레이블(예: DD)을 이 장치로 설정합니다.
3. Kickstart 파일에 다음 행을 추가합니다.

```
driverdisk LABEL=DD:/e1000.rpm
```

DD를 특정 레이블로 바꾸고 e1000.rpm을 특정 이름으로 교체합니다. 하드 디스크 드라이브를 지정하려면 LABEL 대신 inst.repo 명령에서 지원하는 모든 항목을 사용하십시오.

B.2.4. eula

eula Kickstart 명령은 선택 사항입니다. 이 옵션을 사용하여 사용자 개입 없이 EULA(최종 사용자 라이센스 계약)를 수락합니다. 이 옵션을 지정하면 설치를 완료하고 시스템을 처음으로 재부팅한 후 Initial Setup에서 라이센스 계약을 수락하라는 메시지가 표시되지 않습니다.

구문

```
eula [--agreed]
```

옵션

- ***--agreed (필수) - EULA**를 수락합니다. 이 옵션을 항상 사용해야 합니다. 그러지 않으면 **eula** 명령은 의미가 없습니다.*

B.2.5. firstboot

***firstboot Kickstart** 명령은 선택 사항입니다. 시스템을 처음 부팅할 때 **Initial Setup** 애플리케이션이 시작되는지 여부를 결정합니다. 활성화된 경우 **initial-setup** 패키지가 설치되어 있어야 합니다. 지정하지 않으면 이 옵션은 기본적으로 비활성화되어 있습니다.*

구문

```
firstboot OPTIONS
```

옵션

- ***--enable** 또는 **--enabled** - 시스템을 처음 부팅할 때 **Initial Setup**이 시작됩니다.*
- ***--disable** 또는 **--disabled** - 시스템을 처음 부팅할 때 **Initial Setup**이 시작되지 않습니다.*
- ***--reconfig** - 재구성 모드에서 부팅 시 **Initial Setup**을 시작할 수 있습니다. 이 모드는 기본 암호 외에도 루트 암호, 시간 및 날짜 및 네트워킹 및 호스트 이름 구성 옵션을 활성화합니다.*

B.2.6. graphical

***graphical Kickstart** 명령은 선택 사항입니다. 그래픽 모드에서 설치를 수행합니다. 이는 기본값입니다.*

구문

```
graphical [--non-interactive]
```

옵션

- **--non-interactive -** 완전히 비대화형 모드로 설치를 수행합니다. 이 모드는 사용자 상호 작용이 필요한 경우 설치를 종료합니다.

참고

- 완전히 자동 설치의 경우 사용 가능한 모드(**graphical**,**text** 또는 **cmdline**) 중 하나를 **Kickstart** 파일에서 지정해야 합니다. 그렇지 않으면 **console=** 부팅 옵션을 사용해야 합니다. 모드가 지정되지 않은 경우 시스템은 그래픽 모드를 사용하거나 **VNC** 및 텍스트 모드에서 선택하라는 메시지를 표시합니다.

B.2.7. halt

halt Kickstart 명령은 선택 사항입니다.

설치가 성공적으로 완료된 후 시스템을 중지합니다. 이는 수동 설치와 유사합니다. **Anaconda**에서 메시지를 표시하고 사용자가 재부팅하기 전에 키를 누를 때까지 기다립니다. **Kickstart** 설치 중에 완료 방법이 지정되지 않은 경우 이 옵션이 기본값으로 사용됩니다.

구문

```
halt
```

참고

- **halt** 명령은 **shutdown -H** 명령과 동일합니다. 자세한 내용은 **shutdown(8)** 매뉴얼 페이지를 참조하십시오.

- *다른 완료 방법은 **poweroff,reboot, shutdown** 명령을 참조하십시오.*

- *이 명령에는 옵션이 없습니다.*

B.2.8. harddrive

***harddrive Kickstart** 명령은 선택 사항입니다. 로컬 드라이브의 **Red Hat** 설치 트리 또는 전체 설치 **ISO** 이미지에서 설치를 수행합니다. 드라이브는 설치 프로그램이 마운트할 수 있는 파일 시스템인 **ext2, ext3, ext4, vfat, xfs**로 포맷해야 합니다.*

구문

```
harddrive OPTIONS
```

옵션

- ***--partition= -** 설치할 파티션 (예: **sdb2**).*

- ***--dir= -** 설치 트리의 변형 디렉터리 또는 전체 설치 **DVD**의 **ISO** 이미지가 포함된 디렉터리입니다.*

예제

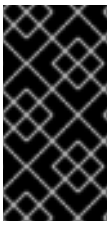
```
harddrive --partition=hdb2 --dir=/tmp/install-tree
```

참고

- *이전에는 **harddrive** 명령을 **install** 명령과 함께 사용해야 했습니다. **install** 명령이 더 이상 사용되지 않으며 하드 드라이브 는 설치를 의미하는 이므로 자체적으로 사용할 수 있습니다.*

- *실제로 설치를 실행하려면 **cdrom,hard drive,hmc,nfs,liveimg** 또는 **url** 중 하나를 지정해야 합니다.*

B.2.9. 설치(더 이상 사용되지 않음)



중요

***Red Hat Enterprise Linux 8**에서는 **install Kickstart** 명령이 더 이상 사용되지 않습니다. 해당 메서드를 별도의 명령으로 사용합니다.*

***Kickstart** 설치 명령은 선택 사항입니다. 기본 설치 모드를 지정합니다.*

구문

```
install
installation_method
```

참고

- *설치 명령 뒤에 설치 방법 명령이 와야 합니다. 설치 방법 명령은 별도의 줄에 있어야 합니다.*
- *이 방법은 다음과 같습니다.*
  - ***cdrom***
  - ***harddrive***
  - ***hmc***
  - ***nfs***

- *liveimg*
- *url*

메서드에 대한 자세한 내용은 별도의 참조 페이지를 참조하십시오.

B.2.10. liveimg

***liveimg Kickstart** 명령은 선택 사항입니다. 패키지 대신 디스크 이미지에서 설치를 수행합니다.*

구문

```
liveimg --url=SOURCE [OPTIONS]
```

필수 옵션

- ***--URL= -** 설치할 위치입니다. 지원되는 프로토콜은 **HTTP, HTTPS, FTP, file**입니다.*

선택적 옵션

- ***--URL= -** 설치할 위치입니다. 지원되는 프로토콜은 **HTTP, HTTPS, FTP, file**입니다.*
- ***--proxy= -** 설치를 수행하는 동안 사용할 **HTTP,HTTPS** 또는 **FTP** 프록시를 지정합니다.*
- ***--checksum= -** 확인에 사용되는 이미지 파일의 **SHA256** 체크섬이 있는 선택적 인수입니다.*
- ***--noverifyssl - HTTPS** 서버에 연결할 때 **SSL** 확인을 비활성화합니다.*

예제

```
liveimg --url=file:///images/install/squashfs.img --
checksum=03825f567f17705100de3308a20354b4d81ac9d8bed4bb4692b2381045e56197 --
noverifyssl
```

참고

- 이미지는 라이브 **ISO** 이미지의 **squashfs.img** 파일, 압축된 **tar** 파일(**.tar .tbz,.tgz,.txz,.tar.bz2,. tar.gz.)** 또는 설치 미디어를 마운트할 수 있는 파일 시스템일 수 있습니다. 지원되는 파일 시스템은 **ext2,ext3,ext4,vfat, xfs**입니다.

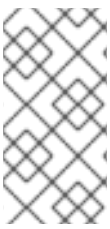
- 드라이버 디스크와 함께 **liveimg** 설치 모드를 사용하면 디스크의 드라이버가 설치된 시스템에 자동으로 포함되지 않습니다. 필요한 경우 이러한 드라이버를 수동으로 설치하거나 **kickstart** 스크립트의 **%post** 섹션에 설치해야 합니다.

- 실제로 설치를 실행하려면 **cdrom,hard drive,hmc,nfs,liveimg** 또는 **url** 중 하나를 지정해야 합니다.

- 이전에는 **liveimg** 명령을 **install** 명령과 함께 사용해야 했습니다. **install** 명령이 더 이상 사용되지 않으며 **liveimg** 는 설치를 의미하기 때문에 자체적으로 사용할 수 있습니다.

B.2.11. logging

logging Kickstart 명령은 선택 사항입니다. 설치 중에 **Anaconda**의 로깅 오류를 제어합니다. 이는 설치된 시스템에 영향을 미치지 않습니다.



참고

로깅은 **TCP**에서만 지원됩니다. 원격 로깅의 경우 **--port=** 옵션에 지정하는 포트 번호가 원격 서버에서 열려 있는지 확인합니다. 기본 포트는 **514**입니다.

구문

```
logging OPTIONS
```

선택적 옵션

- **--host= -** 원격 로깅을 수락하도록 구성된 **syslogd** 프로세스를 실행해야 하는 지정된 원격 호스트에 로깅 정보를 보냅니다.

- **--port= -** 원격 **syslogd** 프로세스에서 기본값 이외의 포트를 사용하는 경우 이 옵션을 사용하여 설정합니다.

- **--level= - tty3**에 표시되는 최소 수준의 메시지를 지정합니다. 그러나 모든 메시지는 이 수준에 관계없이 로그 파일로 전송됩니다. 가능한 값은 **debug,info,warning,error, critical** 입니다.

B.2.12. mediacheck

mediacheck Kickstart 명령은 선택 사항입니다. 이 명령은 설치를 시작하기 전에 설치 프로그램이 미디어 검사를 수행하도록 강제합니다. 이 명령을 실행하려면 설치에 참여해야 하므로 기본적으로 비활성화되어 있습니다.

구문

```
mediacheck
```

참고

- 이 **Kickstart** 명령은 **rd.live.check** 부팅 옵션과 동일합니다.

- 이 명령에는 옵션이 없습니다.

B.2.13. nfs

nfs Kickstart 명령은 선택 사항입니다. 지정된 **NFS** 서버에서 설치를 수행합니다.

구문

```
nfs OPTIONS
```

옵션

- **--server= -** 설치할 서버(호스트 이름 또는 **IP**)입니다.
- **--dir= -** 설치 트리의 변형 디렉터리가 포함된 디렉터리입니다.
- **--opts= - NFS** 내보내기를 마운트하는 데 사용할 마운트 옵션입니다. (선택 사항)

예제

```
nfs --server=nfsserver.example.com --dir=/tmp/install-tree
```

참고

- 이전에는 **nfs** 명령을 **install** 명령과 함께 사용해야 했습니다. **install** 명령은 더 이상 사용되지 않으며 **install** 을 의미하기 때문에 **nfs** 를 자체적으로 사용할 수 있습니다.
- 실제로 설치를 실행하려면 **cdrom,hard drive,hmc,nfs,liveimg** 또는 **url** 중 하나를 지정해야 합니다.

B.2.14. ostreesetup

ostreesetup Kickstart 명령은 선택 사항입니다. 이는 **OStree** 기반 설치를 설정하는 데 사용됩니다.

구문

```
ostreesetup --osname=OSNAME [--remote=REMOTE] --url=URL --ref=REF [--nogpg]
```

필수 옵션:

- **--osname=OSNAME - OS** 설치를 위한 관리 루트입니다.
- **--URL=URL -** 설치할 리포지토리의 **URL**입니다.
- **--ref=REF -** 설치에 사용할 저장소의 분기 이름입니다.

선택적 옵션:

- **--remote=REMOTE - OS** 설치를 위한 관리 루트.
- **--nogpg - GPG** 키 확인을 비활성화합니다.

참고

- **OStree** 툴에 대한 자세한 내용은 업스트림 문서를 참조하십시오. https://ostree.readthedocs.io/en/latest/

B.2.15. poweroff

poweroff Kickstart 명령은 선택 사항입니다. 설치가 완료되면 시스템을 종료하고 전원을 끕니다. 일반적으로 수동 설치 중에 **Anaconda**는 메시지를 표시하고 사용자가 재부팅하기 전에 키를 누를 때까지 기다립니다.

구문

```
poweroff
```

참고

- **poweroff** 옵션은 **shutdown -P** 명령과 동일합니다. 자세한 내용은 **shutdown(8)** 매뉴얼 페이지를 참조하십시오.
- 기타 완료 방법은 **halt, reboot, shutdown Kickstart** 명령을 참조하십시오. **Kickstart** 파일에 다른 방법이 명시적으로 지정되지 않은 경우 **halt** 옵션은 기본 완료 방법입니다.
- **poweroff** 명령은 사용 중인 시스템 하드웨어에 따라 크게 달라집니다. 특히 **BIOS, APM(**고급 전원 관리**)** 및 **ACPI(**고급 구성 및 전원 인터페이스**)**와 같은 특정 하드웨어 구성 요소가 시스템 커널과 상호 작용할 수 있어야 합니다. 시스템의 **APM/ACPI** 기능에 대한 자세한 내용은 하드웨어 설명서를 참조하십시오.
- 이 명령에는 옵션이 없습니다.

B.2.16. reboot

reboot Kickstart 명령은 선택 사항입니다. 설치가 성공적으로 완료된 후 설치 프로그램이 재부팅되도록 지시합니다(개체 없음). 일반적으로 **Kickstart**는 메시지를 표시하고 사용자가 재부팅하기 전에 키를 누를 때까지 기다립니다.

구문

```
reboot OPTIONS
```

옵션

- *--eject - 재부팅하기 전에 부팅 가능한 미디어(DVD, USB 또는 기타 미디어)를 비우십시오.*

- *--kexec - 전체 재부팅 대신 kexec 시스템 호출을 사용하여 설치된 시스템을 메모리에 즉시 로드하여 BIOS 또는 펌웨어에서 일반적으로 수행하는 하드웨어 초기화를 건너뜁니다.*

중요

이 옵션은 더 이상 사용되지 않으며 기술 프리뷰로만 제공됩니다. 기술 프리뷰 기능에 대한 Red Hat 지원 범위 정보는 기술 프리뷰 기능 지원 범위 문서를 참조하십시오.

kexec 를 사용하면 (일반적으로 전체 시스템 재부팅 중에 지워지는 장치 레지스터)가 데이터로 채워질 수 있으며 일부 장치 드라이버에 대한 문제가 발생할 수 있습니다.

참고

- *재부팅 옵션을 사용하면 설치 미디어 및 방법에 따라 끝없는 설치 루프가 발생할 수 있습니다.*

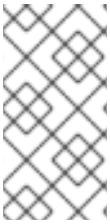
- *reboot 옵션은 shutdown -r 명령과 동일합니다. 자세한 내용은 shutdown(8) 매뉴얼 페이지를 참조하십시오.*

- *64비트 IBM Z에 명령줄 모드로 설치할 때 설치를 완전히 자동화하려면 재부팅 을 지정합니다.*

- *기타 완료 방법은 halt,poweroff, shutdown Kickstart 옵션을 참조하십시오. Kickstart 파일에 다른 방법이 명시적으로 지정되지 않은 경우 halt 옵션은 기본 완료 방법입니다.*

B.2.17. rhsm

rhsm Kickstart 명령은 선택 사항입니다. CDN에서 RHEL을 등록하고 설치하도록 설치 프로그램에 지시합니다.



참고

rhsm Kickstart 명령은 시스템을 등록할 때 사용자 지정 **%post** 스크립트를 사용하는 요구 사항을 제거합니다.

옵션

- **--organization= - CDN**에서 **RHEL**을 등록하고 설치하려면 조직 **ID**를 사용합니다.
- **--activation-key= -** 활성화 키를 사용하여 **CDN**에서 **RHEL**을 등록하고 설치합니다. 이 활성화 키가 서브스크립션에 등록된 경우 활성화 키당 한 번 옵션을 여러 번 사용할 수 있습니다.
- **--connect-to-insights -** 대상 시스템을 **Red Hat Insights**에 연결합니다.
- **--proxy= - HTTP** 프록시를 설정합니다.
- **--server-hostname= - Satellite** 인스턴스 **URL**을 설정합니다. **Red Hat** 서브스크립션 인프라에 대신 **Satellite** 인스턴스에 등록하려면 이 옵션을 사용합니다.

B.2.18. shutdown

shutdown Kickstart 명령은 선택 사항입니다. 설치가 성공적으로 완료된 후 시스템이 종료됩니다.

구문

```
shutdown
```

참고

- **shutdown Kickstart** 옵션은 **shutdown** 명령과 동일합니다. 자세한 내용은 **shutdown(8)** 매뉴얼 페이지를 참조하십시오.
-

다른 완료 방법의 경우 **halt, poweroff, reboot Kickstart** 옵션을 참조하십시오. **Kickstart** 파일에 다른 방법이 명시적으로 지정되지 않은 경우 **halt** 옵션은 기본 완료 방법입니다.

- 이 명령에는 옵션이 없습니다.

B.2.19. sshpw

sshpw Kickstart 명령은 선택 사항입니다.

설치하는 동안 설치 프로그램과 상호 작용하고 **SSH** 연결을 통해 진행 상황을 모니터링할 수 있습니다. **sshpw** 명령을 사용하여 로그인할 임시 계정을 만듭니다. 명령의 각 인스턴스는 설치 환경에만 존재하는 별도의 계정을 생성합니다. 이러한 계정은 설치된 시스템으로 전송되지 않습니다.

구문

```
sshpw --username=name [OPTIONS] password
```

필수 옵션

- **--username=name -** 사용자 이름을 제공합니다. 이 옵션은 필수입니다.
- **password -** 사용자에게 사용할 암호입니다. 이 옵션은 필수입니다.

선택적 옵션

- **--iscrypted -** 이 옵션이 있는 경우 암호 인수가 이미 암호화된 것으로 간주됩니다. 이 옵션은 **--plaintext** 와 함께 사용할 수 없습니다. 암호화된 암호를 만들려면 **Python**을 사용할 수 있습니다.

```
$ python3 -c 'import crypt,getpass;pw=getpass.getpass();print(crypt.crypt(pw) if
(pw==getpass.getpass("Confirm: ")) else exit())'
```

그러면 임의의 **Salt**를 사용하여 암호의 **sha512 crypt** 호환 해시가 생성됩니다.

- ***--plaintext -** 이 옵션이 있는 경우 **password** 인수는 일반 텍스트로 간주됩니다. 이 옵션은 **--iscrypted**와 함께 사용할 수 없습니다.*

- ***--lock -** 이 옵션이 있는 경우 이 계정은 기본적으로 잠깁니다. 즉, 사용자가 콘솔에서 로그인할 수 없습니다.*

- ***--sshkey -** 옵션이 있는 경우 **<password>** 문자열이 **ssh** 키 값으로 해석됩니다.*

참고

- *기본적으로 설치 중에 **ssh** 서버가 시작되지 않습니다. 설치 중에 **ssh** 를 사용할 수 있도록 하려면 커널 부팅 옵션 **inst.sshd** 를 사용하여 시스템을 부팅합니다.*

- ***root ssh** 액세스를 비활성화하려면 다른 사용자 **ssh** 액세스를 허용하는 동안 다음을 사용하십시오.*

```
sshpw --username=example_username example_password --plaintext
sshpw --username=root example_password --lock
```

- ***root ssh** 액세스를 비활성화하려면 다음을 사용하십시오.*

```
sshpw --username=root example_password --lock
```

B.2.20. text

***text Kickstart** 명령은 선택 사항입니다. **Kickstart** 설치를 텍스트 모드로 수행합니다. **Kickstart** 설치는 기본적으로 그래픽 모드에서 수행됩니다.*

구문

```
text [--non-interactive]
```

옵션

- *--non-interactive - 완전히 비대화형 모드로 설치를 수행합니다. 이 모드는 사용자 상호 작용이 필요한 경우 설치를 종료합니다.*

참고

- *완전 자동 설치의 경우 사용 가능한 모드(graphical,text 또는 cmdline)를 Kickstart 파일에서 지정해야 합니다. 그렇지 않으면 console= 부팅 옵션을 사용해야 합니다. 모드가 지정되지 않은 경우 시스템은 그래픽 모드를 사용하거나 VNC 및 텍스트 모드에서 선택하라는 메시지를 표시합니다.*

B.2.21. url

url Kickstart 명령은 선택 사항입니다. FTP, HTTP 또는 HTTPS 프로토콜을 사용하여 원격 서버의 설치 트리 이미지에서 설치하는 데 사용됩니다. 하나의 URL만 지정할 수 있습니다.

구문

```
url --url=FROM [OPTIONS]
```

필수 옵션

- *--URL=FROM - 설치할 HTTP,HTTPS,FTP 또는 file 위치를 지정합니다.*

선택적 옵션

- *--mirrorlist= - 설치할 미러 URL을 지정합니다.*
- *--proxy= - 설치 중에 사용할 HTTP,HTTPS 또는 FTP 프록시를 지정합니다.*
- *--noverifyssl - HTTPS 서버에 연결할 때 SSL 확인을 비활성화합니다.*
- *--Metalink=URL - 설치할 metalink URL을 지정합니다. URL의 $releasever 및 $basearch에 대해 변수 대체가 수행됩니다.*

예제

- ***HTTP** 서버에서 설치하려면 다음을 수행합니다.*

```
url --url=http://server/path
```

- ***FTP** 서버에서 설치하려면 다음을 수행합니다.*

```
url --url=ftp://username:password@server/path
```

- *로컬 파일에서 설치하려면 다음을 수행합니다.*

```
liveimg --url=file:///images/install/squashfs.img --noverifyssl
```

참고

- *이전에는 **install** 명령과 함께 **url** 명령을 사용해야 했습니다. **install** 명령은 더 이상 사용되지 않으며 **url**은 **install**을 의미하기 때문에 자체적으로 사용할 수 있습니다.*

- *실제로 설치를 실행하려면 **cdrom,hard drive,hmc,nfs,liveimg** 또는 **url** 중 하나를 지정해야 합니다.*

B.2.22. vnc

***vnc Kickstart** 명령은 선택 사항입니다. **VNC**를 통해 그래픽 설치를 원격으로 볼 수 있습니다.*

*일반적으로 이 방법은 텍스트 설치에 일부 크기 및 언어 제한이 있으므로 텍스트 모드에서 기본 설정됩니다. 추가 옵션이 없으면 이 명령은 암호 없이 설치 시스템에서 **VNC** 서버를 시작하고 연결하는 데 필요한 세부 정보를 표시합니다.*

구문

```
vnc [--host=host_name] [--port=port] [--password=password]
```

옵션

--host=

지정된 호스트 이름에서 수신 대기하는 VNC 뷰어 프로세스에 연결합니다.

--port=

원격 VNC 뷰어 프로세스가 수신 대기 중인 포트를 제공합니다. 제공되지 않는 경우 Anaconda 는 VNC 기본 포트 5900을 사용합니다.

--password=

VNC 세션에 연결하기 위해 제공해야 하는 암호를 설정합니다. 이는 선택 사항이지만 권장됩니다.

추가 리소스

- PXE를 사용하여 네트워크에서 설치 준비

B.2.23. %include

%include Kickstart 명령은 선택 사항입니다.

내용이 Kickstart 파일에 있는 **%include** 명령의 위치에 있는 것처럼 Kickstart 파일에 다른 파일의 내용을 포함하려면 **%include** 명령을 사용합니다.

이러한 포함은 **%pre** 스크립트 섹션 이후에만 평가되므로 **%pre** 섹션의 스크립트에서 생성된 파일을 포함하는 데 사용할 수 있습니다. **%pre** 섹션을 평가하기 전에 파일을 포함하려면 **%ksappend** 명령을 사용합니다.

구문

```
%include path/to/file
```

B.2.24. %ksappend

%ksappend Kickstart 명령은 선택 사항입니다.

내용이 **Kickstart** 파일에서 **%ksappend** 명령의 위치에 있는 것처럼 **Kickstart** 파일에 다른 파일의 내용을 포함하도록 **%ksappend** 명령을 사용합니다.

이러한 포함은 **%include** 명령과 달리 **%pre** 스크립트 섹션보다 먼저 평가됩니다.

구문

```
%ksappend path/to/file
```

B.3. 시스템 구성을 위한 KICKSTART 명령

이 목록의 **Kickstart** 명령은 사용자, 리포지토리 또는 서비스와 같은 결과 시스템에 대한 자세한 내용을 구성합니다.

B.3.1. auth 또는 authconfig(더 이상 사용되지 않음)

중요

더 이상 사용되지 않는 **auth** 또는 **authconfig Kickstart** 명령 대신 새 **authselect** 명령을 사용합니다. **auth** 및 **authconfig** 는 제한된 이전 버전과의 호환성을 위해서만 사용할 수 있습니다.

auth 또는 **authconfig Kickstart** 명령은 선택 사항입니다. **authconfig** 도구를 사용하여 시스템의 인증 옵션을 설정합니다. 이 옵션은 설치가 완료된 후에도 명령줄에서 실행할 수 있습니다.

구문

```
authconfig [OPTIONS]
```

참고

- *이전에는 **authconfig** 툴이라는 **auth** 또는 **authconfig Kickstart** 명령도 있었습니다. 이 툴은 **Red Hat Enterprise Linux 8**에서 더 이상 사용되지 않습니다. 이러한 **Kickstart** 명령은 이제 **authselect-compat** 툴을 사용하여 새 **authselect** 툴을 호출합니다. 호환성 계층 및 알려진 문제에 대한 설명은 수동 페이지 **authselect-migration(7)** 을 참조하십시오. 설치 프로그램은 더 이상 사용되지 않는 명령의 사용을 자동으로 감지하고 호환성 계층을 제공하기 위해 **authselect-compat** 패키지를 시스템에 설치합니다.*

- *암호는 기본적으로 새도우됩니다.*

- *보안을 위해 **OpenLDAP**를 **SSL** 프로토콜로 사용하는 경우 서버 구성에서 **SSLv2** 및 **SSLv3** 프로토콜이 비활성화되어 있는지 확인합니다. 이는 **POODLE SSL** 취약점(**CVE-2014-3566**) 때문입니다. 자세한 내용은 **https://access.redhat.com/solutions/1234843** 를 참조하십시오.*

B.3.2. Authselect

***authselect Kickstart** 명령은 선택 사항입니다. **authselect** 명령을 사용하여 시스템에 대한 인증 옵션을 설정합니다. 이는 설치가 완료된 후 명령줄에서도 실행할 수 있습니다.*

구문

```
authselect [OPTIONS]
```

참고

- *이 명령은 모든 옵션을 **authselect** 명령에 전달합니다. 자세한 내용은 **authselect(8)** 매뉴얼 페이지 및 **authselect --help** 명령을 참조하십시오.*

- *이 명령은 **Red Hat Enterprise Linux 8**에서 더 이상 사용되지 않는 **auth** 또는 **authconfig** 명령을 **authconfig** 툴과 함께 대체합니다.*

- 암호는 기본적으로 새도우됩니다.

- 보안을 위해 **OpenLDAP**를 **SSL** 프로토콜로 사용하는 경우 서버 구성에서 **SSLv2** 및 **SSLv3** 프로토콜이 비활성화되어 있는지 확인합니다. 이는 **POODLE SSL** 취약점(**CVE-2014-3566**) 때문입니다. 자세한 내용은 https://access.redhat.com/solutions/1234843 를 참조하십시오.

B.3.3. 방화벽

firewall Kickstart 명령은 선택 사항입니다. 설치된 시스템의 방화벽 구성을 지정합니다.

구문

```
firewall --enabled|--disabled [incoming] [OPTIONS]
```

필수 옵션

- **--enabled** 또는 **--enable - DNS** 응답 또는 **DHCP** 요청과 같은 아웃바운드 요청에 응답하지 않는 들어오는 연결을 다시 생성합니다. 이 시스템에서 실행 중인 서비스에 대한 액세스가 필요한 경우 방화벽을 통해 특정 서비스를 허용하도록 선택할 수 있습니다.

- **--disabled** 또는 **--disable - iptables** 규칙을 구성하지 않습니다.

선택적 옵션

- **--trust - em1** 과 같은 장치를 나열하면 해당 장치로 들어오는 모든 트래픽이 방화벽을 통과할 수 있습니다. 둘 이상의 장치를 나열하려면 **--trust em1 --trust em2** 와 같은 옵션을 더 사용합니다. **--trust em1, em2** 와 같은 쉼표로 구분된 형식을 사용하지 마십시오.

- **--remove-service -** 방화벽을 통해 서비스를 허용하지 않습니다.

- 들어오는 **-** 방화벽을 통해 지정된 서비스를 허용하기 위해 다음 중 하나 이상으로 바꿉니다.

- ***--ssh***
- ***--smtp***
- ***--http***
- ***--ftp***

- ***--port= - port:protocol** 형식을 사용하여 방화벽을 통해 포트를 허용하도록 지정할 수 있습니다. 예를 들어 방화벽을 통한 **IMAP** 액세스를 허용하려면 **InstallPlan :tcp**를 지정합니다. 숫자 포트도 명시적으로 지정할 수 있습니다. 예를 들어 포트 **1234**에서 **UDP** 패킷을 허용하도록 **1234:udp** 를 지정합니다. 여러 포트를 지정하려면 쉼표로 구분합니다.*

- ***--service= -** 이 옵션은 방화벽을 통해 서비스를 허용하는 상위 수준의 방법을 제공합니다. 일부 서비스(예: **cups,avahi** 등)에는 서비스가 작동하려면 여러 개의 포트가 열려 있거나 기타 특수 구성이 필요합니다. **--port** 옵션을 사용하여 각 개별 포트를 지정하거나 **--service=** 를 지정하여 한 번에 모두 열 수 있습니다.*

  *유효한 옵션은 **firewalld** 패키지에서 **firewall-offline-cmd** 프로그램으로 인식됩니다. **firewalld** 서비스가 실행 중인 경우 **firewall-cmd --get-services** 는 알려진 서비스 이름 목록을 제공합니다.*

- ***--use-system-defaults -** 방화벽을 전혀 구성하지 마십시오. 이 옵션은 **anaconda**에 아무 작업도 수행하지 않으며 시스템이 패키지 또는 **ostree**와 함께 제공된 기본값을 사용하도록 지시합니다. 이 옵션을 다른 옵션과 함께 사용하면 다른 모든 옵션이 무시됩니다.*

B.3.4. group

*그룹 **Kickstart** 명령은 선택 사항입니다. 시스템에 새 사용자 그룹을 생성합니다.*

```
group --name=name [--gid=gid]
```

필수 옵션

- ***--name= -** 그룹의 이름을 제공합니다.*

선택적 옵션

- **--GID= -** 그룹의 **GID**입니다. 제공하지 않는 경우 기본값은 사용 가능한 다음 시스템 이외의 **GID**로 설정됩니다.

참고

- 지정된 이름 또는 **GID**가 있는 그룹이 이미 있는 경우 이 명령이 실패합니다.
- **user** 명령을 사용하여 새로 만든 사용자에 대한 새 그룹을 만들 수 있습니다.

B.3.5. 키보드(필수)

키보드 **Kickstart** 명령이 필요합니다. 시스템에 대해 사용 가능한 하나 이상의 키보드 레이아웃을 설정합니다.

구문

```
keyboard --vckeymap|--xlayouts OPTIONS
```

옵션

- **--vckeymap= -** 사용할 **VConsole** 키맵을 지정합니다. 유효한 이름은 **.map.gz** 확장자 없이 **/usr/lib/kbd/keymaps/xkb/** 디렉터리의 파일 목록에 해당합니다.
- **--xlayouts= -** 공백 없이 쉼표로 구분된 목록으로 사용해야 하는 **X** 레이아웃 목록을 지정합니다. 레이아웃 형식(예: **cz**) 또는 레이아웃 (예: **cz**) 형식(예: **c werty**) 과 같은 **setxkbmap(1)**형식의 값을 허용합니다.

  사용 가능한 모든 레이아웃은 레이아웃 아래의 **xkeyboard-config(7)** 도움말 페이지에서 볼 수 있습니다.
- **--switch= - layout-switching** 옵션 목록을 지정합니다(여러 키보드 레이아웃 간에 전환하기 위한 단축). 여러 옵션은 공백 없이 쉼표로 구분해야 합니다. **setxkbmap(1)** 과 동일한 형식의

값을 허용합니다.

*사용 가능한 전환 옵션은 옵션 아래의 **xkeyboard-config(7)** 도움말 페이지에서 볼 수 있습니다.*

참고

- ***--vckeymap=** 또는 **--xlayouts=** 옵션을 사용해야 합니다.*

예제

*다음 예제에서는 **--xlayouts=** 옵션을 사용하여 두 개의 키보드 레이아웃 **(US)** 및 체코 **(qwerty)**를 설정하고 **Alt+Shift** 를 사용하여 전환할 수 있습니다.*

```
keyboard --xlayouts=us,'cz (qwerty)' --switch=grp:alt_shift_toggle
```

B.3.6. lang(필수)

***lang Kickstart** 명령이 필요합니다. 설치 중 사용할 언어와 설치된 시스템에서 사용할 기본 언어를 설정합니다.*

구문

```
lang language [--addsupport=language,...]
```

필수 옵션

- ***Language - Install support for this language and set it as system default.***

선택적 옵션

- ***--addsupport= -** 추가 언어에 대한 지원 추가. 공백 없이 쉼표로 구분된 목록 형식을 사용합니다. 예를 들어 다음과 같습니다.*

```
lang en_US --addsupport=cs_CZ,de_DE,en_UK
```

참고

- **locale -a | grep _** 또는 **localectl list-locales | grep _** 명령은 지원되는 로케일 목록을 반환합니다.

- 특정 언어(예: 중국어, 일본어, 한국어)는 텍스트 모드 설치 중에 지원되지 않습니다. **lang** 명령을 사용하여 이러한 언어 중 하나를 지정하면 설치 프로세스가 영어로 진행되지만 설치된 시스템은 선택을 기본 언어로 사용합니다.

예제

언어를 **English**로 설정하려면 **Kickstart** 파일에 다음 행이 포함되어야 합니다.

```
lang en_US
```

B.3.7. module

모듈 **Kickstart** 명령은 선택 사항입니다. 이 명령을 사용하여 **Kickstart** 스크립트 내에서 패키지 모듈 스트림을 활성화합니다.

구문

```
module --name=NAME [--stream=STREAM]
```

필수 옵션

--name=

활성화할 모듈의 이름을 지정합니다. **NAME** 을 실제 이름으로 바꿉니다.

선택적 옵션

--stream=

활성화할 모듈 스트림의 이름을 지정합니다. **STREAM** 을 실제 이름으로 바꿉니다.

기본 스트림이 정의된 모듈에 이 옵션을 지정할 필요가 없습니다. 기본 스트림이 없는 모듈의 경

우 이 옵션은 필수이며 비워 두면 오류가 발생합니다. 다양한 스트림을 사용하여 모듈을 여러 번 활성화할 수 없습니다.

참고

- 이 명령과 **%packages** 섹션을 사용하면 모듈 및 스트림을 명시적으로 지정하지 않고 활성화된 모듈과 스트림 조합에서 제공하는 패키지를 설치할 수 있습니다. 패키지를 설치하기 전에 모듈을 활성화해야 합니다. **module** 명령을 사용하여 모듈을 활성화한 후 **%packages** 섹션에 나열하여 이 모듈에서 활성화한 패키지를 설치할 수 있습니다.

- 단일 **module** 명령은 단일 모듈 및 스트림 조합만 활성화할 수 있습니다. 여러 모듈을 활성화하려면 여러 모듈 명령을 사용합니다. 다양한 스트림을 사용하여 모듈을 여러 번 활성화할 수 없습니다.

- **Red Hat Enterprise Linux 8**에서는 모듈은 **AppStream** 리포지토리에만 제공됩니다. 사용 가능한 모듈을 나열하려면 유효한 서브스크립션이 설치된 **Red Hat Enterprise Linux 8** 시스템에서 **yum module list** 명령을 사용합니다.

추가 리소스

- 사용자 공간 구성 요소 설치, 관리 및 제거

B.3.8. 리포지토리

repo Kickstart 명령은 선택 사항입니다. 패키지 설치 소스로 사용할 수 있는 추가 **yum** 리포지토리를 구성합니다. 여러 리포지토리 행을 추가할 수 있습니다.

구문

```
repo --name=repoid [--baseurl=url|--mirrorlist=url|--metalink=url] [OPTIONS]
```

필수 옵션

- **--name= -** 리포지토리 **ID**입니다. 이 옵션은 필수입니다. 리포지토리에 이전에 추가한 다른 리포지토리와 충돌하는 이름이 있으면 무시됩니다. 설치 프로그램은 사전 정의된 리포지토리 목

록을 사용하므로 사전 설정된 이름과 동일한 이름의 리포지토리를 추가할 수 없습니다.

URL 옵션

*이러한 옵션은 상호 배타적이며 선택 사항입니다. **yum** 리포지토리 구성 파일에서 사용할 수 있는 변수는 여기에서 지원되지 않습니다. **$releasever** 및 **$basearch** 문자열을 **URL**의 각 값으로 교체할 수 있습니다.*

- ***--BASEURL= -** 리포지토리의 **URL**입니다.*
- ***--mirrorlist= -** 저장소에 대한 미러 목록을 가리키는 **URL**입니다.*
- ***--Metalink= -** 리포지토리의 **metalink**가 있는 **URL**입니다.*

선택적 옵션

- ***--install -** 설치된 시스템에 제공된 리포지토리 구성을 **/etc/yum.repos.d/** 디렉토리에 저장합니다. 이 옵션을 사용하지 않으면 **Kickstart** 파일에 구성된 리포지토리는 설치된 시스템이 아니라 설치 프로세스 중에만 사용할 수 있습니다.*
- ***--cost= -** 이 리포지토리에 비용을 할당하는 정수 값입니다. 여러 리포지토리가 동일한 패키지를 제공하는 경우 이 숫자는 다른 리포지토리보다 먼저 사용할 리포지토리 우선 순위를 지정하는 데 사용됩니다. 비용이 낮은 리포지토리는 더 높은 비용이 있는 리포지토리보다 우선합니다.*
- ***--excludepkgs= -** 이 리포지토리에서 가져오지 않아야 하는 쉼표로 구분된 패키지 이름 목록입니다. 이 기능은 여러 리포지토리가 동일한 패키지를 제공하고 특정 리포지토리에서 제공하는지 확인하려는 경우에 유용합니다. 전체 패키지 이름(**예: publican**) 및 **glob**(**예: gnome-***)가 허용됩니다.*
- ***--includepkgs= -** 이 리포지토리에서 가져올 수 있는 쉼표로 구분된 패키지 이름 및 **glob** 목록입니다. 리포지토리에서 제공하는 다른 패키지는 무시됩니다. 이는 리포지토리에서 제공하는 다른 모든 패키지를 제외하면서 리포지토리에서 제공하는 단일 패키지 또는 리포지토리에서 제공하는 패키지 세트만 설치하려면 유용합니다.*
- ***--proxy=[프로토콜://][사용자 이름] [:password]@]호스트[:port] -** 이 리포지토리에 대해서만 사용할 **HTTP/HTTPS/ftp** 프록시를 지정합니다. 이 설정은 다른 리포지토리에 영향을 미치지 않으며, **HTTP** 설치에서 **install.img** 를 가져오는 방법에는 영향을 미치지 않습니다.*

- ***--noverifyssl - HTTPS** 서버에 연결할 때 **SSL** 확인을 비활성화합니다.*

참고

- *설치에 사용되는 리포지토리는 안정적이어야 합니다. 설치가 완료되기 전에 리포지토리를 수정하면 설치에 실패할 수 있습니다.*

B.3.9. rootpw(필수)

***rootpw Kickstart** 명령이 필요합니다. 시스템의 루트 암호를 **password** 인수로 설정합니다.*

구문

```
rootpw [--iscrypted|--plaintext] [--lock] password
```

필수 옵션

- *암호 **-** 암호 사양. 일반 텍스트 또는 암호화된 문자열입니다. 아래 **--iscrypted** 및 **--plaintext** 를 참조하십시오.*

옵션

- ***--iscrypted -** 이 옵션이 있는 경우 암호 인수가 이미 암호화된 것으로 간주됩니다. 이 옵션은 **--plaintext** 와 함께 사용할 수 없습니다. 암호화된 암호를 만들려면 **python**을 사용할 수 있습니다.*

```
$ python -c 'import crypt,getpass;pw=getpass.getpass();print(crypt.crypt(pw) if
(pw==getpass.getpass("Confirm: ")) else exit())'
```

*그러면 임의의 **Salt**를 사용하여 암호의 **sha512 crypt** 호환 해시가 생성됩니다.*

- ***--plaintext -** 이 옵션이 있는 경우 **password** 인수는 일반 텍스트로 간주됩니다. 이 옵션은 **--iscrypted** 와 함께 사용할 수 없습니다.*

- ***--lock*** *- 이 옵션이 있는 경우* ***root*** *계정이 기본적으로 잠깁니다. 즉,* ***root*** *사용자는 콘솔에서 로그인할 수 없습니다. 또한 이 옵션은 그래픽 및 텍스트 기반 수동 설치에서 루트 암호 화면을 비활성화합니다.*

B.3.10. selinux

selinux Kickstart *명령은 선택 사항입니다. 설치된 시스템에* ***SELinux****의 상태를 설정합니다. 기본* ***SELinux*** *정책은 강제 적용 입니다.*

구문

```
selinux [--disabled|--enforcing|--permissive]
```

옵션

--enforcing

강제 적용되는 기본 대상 정책을 사용하여 ***SELinux****를 활성화합니다.*

--permissive

SELinux *정책에 따라 경고를 출력하지만 실제로 정책을 적용하지는 않습니다.*

--disabled

시스템에서 ***SELinux****를 완전히 비활성화합니다.*

추가 리소스

- *SElinux 사용*

B.3.11. services

services Kickstart *명령은 선택 사항입니다. 기본* ***systemd*** *대상에서 실행할 서비스의 기본 집합을 수정합니다. 비활성화된 서비스 목록은 활성화된 서비스 목록보다 먼저 처리됩니다. 따라서 서비스가 두 목록에 모두 표시되면 활성화됩니다.*

구문

```
services [--disabled=list] [--enabled=list]
```

옵션

- **--disabled= -** 쉼표로 구분된 목록에 제공된 서비스를 비활성화합니다.
- **--enabled= -** 쉼표로 구분된 목록에 제공된 서비스를 활성화합니다.

참고

- 서비스 목록에 공백을 포함하지 마십시오. 이 경우 **Kickstart**는 첫 번째 공간까지 서비스만 활성화하거나 비활성화합니다. 예를 들어 다음과 같습니다.

  ```
  services --disabled=auditd, cups,smartd, nfslock
  ```

  이는 **auditd** 서비스만 비활성화합니다. 4개의 서비스를 모두 비활성화하려면 이 항목에 공백을 포함하지 않아야 합니다.

  ```
  services --disabled=auditd,cups,smartd,nfslock
  ```

B.3.12. skipx

skipx Kickstart 명령은 선택 사항입니다. 이 경우 **X**가 설치된 시스템에 구성되지 않습니다.

패키지 선택 옵션에 표시 관리자를 설치하는 경우 이 패키지는 **X** 구성을 만들고 설치된 시스템의 기본값은 **graphical.target** 입니다. 이는 **skipx** 옵션의 영향을 덮어씁니다.

구문

```
skipx
```

참고

- *이 명령에는 옵션이 없습니다.*

B.3.13. sshkey

***sshkey Kickstart** 명령은 선택 사항입니다. 설치된 시스템에 지정된 사용자의 **authorized_keys** 파일에 **SSH** 키를 추가합니다.*

구문

```
sshkey --username=user "ssh_key"
```

필수 옵션

- ***--username= -** 키가 설치될 사용자입니다.*
- ***ssh_key -** 완전한 **SSH** 키 지문입니다. 따옴표로 묶어야 합니다.*

B.3.14. syspurpose

***syspurpose Kickstart** 명령은 선택 사항입니다. 이를 사용하여 설치 후 시스템의 사용 방법을 설명하는 시스템 용도를 설정합니다. 이 정보는 시스템에 올바른 서브스크립션 인타이틀먼트를 적용하는 데 도움이 됩니다.*

구문

```
syspurpose [OPTIONS]
```

옵션

- ***--role= -*** *의도한 시스템 역할을 설정합니다. 사용 가능한 값은 다음과 같습니다.*
  - ***Red Hat Enterprise Linux Server***
  - ***Red Hat Enterprise Linux Workstation***
  - ***Red Hat Enterprise Linux Compute Node***
- ***--SLA= -*** *서비스 수준 계약을 설정합니다. 사용 가능한 값은 다음과 같습니다.*
  - ***Premium***
  - ***Standard***
  - ***Self-Support***
- ***--usage= -*** *의도한 시스템 사용 사용 가능한 값은 다음과 같습니다.*
  - ***Production***
  - ***Disaster Recovery***
  - ***Development/Test***
- ***--Addon= -*** *추가 계층화된 제품 또는 기능을 지정합니다. 이 옵션을 여러 번 사용할 수 있습니다.*

참고

- 공백으로 값을 입력하고 큰따옴표로 묶습니다.

```
syspurpose --role="Red Hat Enterprise Linux Server"
```

- 시스템 용도를 구성하는 것이 권장되지만 이는 **Red Hat Enterprise Linux** 설치 프로그램의 선택적 기능입니다. 설치가 완료된 후 시스템 용도를 활성화하려면 **syspurpose** 명령줄 도구를 사용하여 이를 수행할 수 있습니다.

B.3.15. 시간대 (필수)

시간대 **Kickstart** 명령이 필요합니다. 시스템 시간대를 설정합니다.

구문

```
timezone timezone [OPTIONS]
```

필수 옵션

- **timezone -** 시스템에 설정할 시간대입니다.

선택적 옵션

- **--UTC -** 시스템에서 하드웨어 클록이 **UTC(Greenwich Mean)** 시간으로 설정되어 있다고 가정합니다.
- **--nontp - NTP** 서비스 자동 시작을 비활성화합니다.
- **--ntpservers= -** 공백 없이 쉼표로 구분된 목록으로 사용할 **NTP** 서버 목록을 지정합니다.

참고

Red Hat Enterprise Linux 8에서는 **pytz** 패키지에서 제공하는 **pytz.all_timezones** 목록을 사용하여 시간대 이름을 검증합니다. 이전 릴리스에서는 현재 사용되는 목록의 하위 집합인 **pytz.common_timezones** 에 대해 이름이 검증되었습니다. 그래픽 및 텍스트 모드 인터페이스는 계속 제한된 **pytz.common_timezones** 목록을 사용합니다. **Kickstart** 파일을 사용하여 추가 시간대 정의를 사용해야 합니다.

B.3.16. user

사용자 **Kickstart** 명령은 선택 사항입니다. 시스템에 새 사용자를 생성합니다.

구문

```
user --name=username [OPTIONS]
```

필수 옵션

- **--name= -** 사용자 이름을 제공합니다. 이 옵션은 필수입니다.

선택적 옵션

- **--GECOS= -** 사용자에 대한 **GECOS** 정보를 제공합니다. 이는 쉼표로 구분된 다양한 시스템별 필드의 문자열입니다. 사용자 전체 이름, 사무실 번호 등을 지정하는 데 자주 사용됩니다. 자세한 내용은 **passwd(5)** 매뉴얼 페이지를 참조하십시오.

- **--groups= -** 기본 그룹 외에도 사용자가 속해야 하는 그룹 이름의 쉼표로 구분된 목록입니다. 그룹은 사용자 계정을 생성하기 전에 존재해야 합니다. 그룹 명령을 참조하십시오.

- **--homedir= -** 사용자의 홈 디렉터리입니다. 제공하지 않는 경우 기본값은 **/home/username** 입니다.

- **--lock -** 이 옵션이 있는 경우 이 계정은 기본적으로 잠깁니다. 즉, 사용자가 콘솔에서 로그인할 수 없습니다. 이 옵션은 그래픽 및 텍스트 기반 수동 설치 모두에서 **Create User** 화면을 비활성화합니다.

- *--password= - 새 사용자의 암호입니다. 제공하지 않으면 기본적으로 계정이 잠깁니다.*

- *--iscrypted - 이 옵션이 있는 경우 암호 인수가 이미 암호화된 것으로 간주됩니다. 이 옵션은 --plaintext 와 함께 사용할 수 없습니다. 암호화된 암호를 만들려면 python을 사용할 수 있습니다.*

```
$ python -c 'import crypt,getpass;pw=getpass.getpass();print(crypt.crypt(pw) if
(pw==getpass.getpass("Confirm: ")) else exit())'
```

  그러면 임의의 Salt를 사용하여 암호의 sha512 crypt 호환 해시가 생성됩니다.

- *--plaintext - 이 옵션이 있는 경우 password 인수는 일반 텍스트로 간주됩니다. 이 옵션은 --iscrypted와 함께 사용할 수 없습니다.*

- *--shell= - 사용자의 로그인 셸입니다. 제공하지 않으면 시스템 기본값이 사용됩니다.*

- *--UID= - 사용자의 UID(User ID) 제공하지 않는 경우 기본값은 사용 가능한 다음 시스템 이외의 UID로 설정됩니다.*

- *--GID= - 사용자 그룹에 사용할 GID(그룹 ID)입니다. 제공하지 않는 경우 기본값은 사용 가능한 다음 시스템 이외의 그룹 ID로 설정됩니다.*

참고

- *--uid 및 --gid 옵션을 사용하여 일반 사용자 ID와 해당 기본 그룹을 1000 개 대신 5000 부터 설정하는 것이 좋습니다. 이는 시스템 사용자 및 그룹용으로 예약된 범위, 0-999 이므로 나중에 증가하여 일반 사용자 ID와 중복될 수 있기 때문입니다.*

  설치 후 최소 UID 및 GID 제한을 변경하려면 사용자 생성 시 선택한 UID 및 GID 범위가 자동으로 적용되도록 하려면 기본 시스템 설정 문서의 umask를 사용하여 새 파일의 기본 권한 설정 섹션을 참조하십시오.

- *파일 및 디렉토리는 파일 또는 디렉터리를 생성하는 데 사용되는 애플리케이션에 의해 지정된 다양한 권한으로 생성됩니다. 예를 들어, mkdir 명령은 모든 권한이 활성화된 디렉토리를 생성합니다. 그러나 애플리케이션은 사용자 파일 생성 마스크 설정에서 지정한 대로 새로 생성된 파일에 특정 권한을 부여할 수 없습니다.*

사용자 파일 생성 마스크는 **umask** 명령을 사용하여 제어할 수 있습니다. 새 사용자에 대한 사용자 파일 생성 마스크 의 기본 설정은 설치된 시스템의 **/etc/login.defs** 구성 파일의 **UMASK** 변수로 정의됩니다. 설정되지 않은 경우 기본값은 **022** 입니다. 즉, 애플리케이션이 파일을 생성할 때 기본적으로 파일 소유자 이외의 사용자에게 쓰기 권한을 부여할 수 없습니다. 그러나 이 설정은 다른 설정 또는 스크립트로 재정의할 수 있습니다.

기본 시스템 설정 문서의 **umask** 섹션을 사용하여 새 파일의 기본 권한 설정 섹션에서 자세한 내용을 확인할 수 있습니다.

B.3.17. xconfig

xconfig Kickstart 명령은 선택 사항입니다. **X Window System**을 설정합니다.

구문

```
xconfig [--startxonboot]
```

옵션

- **--startxonboot -** 설치된 시스템에서 그래픽 로그인을 사용합니다.

참고

- **Red Hat Enterprise Linux 8**에는 **KDE** 데스크탑 환경이 포함되어 있지 않으므로 업스트림에 문서화된 **--defaultdesktop=** 을 사용하지 마십시오.

B.4. 네트워크 구성을 위한 KICKSTART 명령

이 목록에 있는 **Kickstart** 명령을 사용하여 시스템에서 네트워킹을 구성할 수 있습니다.

B.4.1. 네트워크(선택 사항)

선택적 네트워크 **Kickstart** 명령을 사용하여 대상 시스템에 대한 네트워크 정보를 구성하고 설치 환경에서 네트워크 장치를 활성화합니다. 첫 번째 네트워크 명령에 지정된 장치가 자동으로 활성화됩니다. 활

성화 옵션을 사용하여 장치를 명시적으로 활성화하도록 요청할 수도 있습니다.

구문

```
network OPTIONS
```

옵션

- ***--activate -** 설치 환경에서 이 장치를 활성화합니다.*

  *이미 활성화된 장치에서 **--activate** 옵션을 사용하는 경우(예: 부팅 옵션으로 구성한 인터페이스) 시스템이 **Kickstart** 파일에 지정된 세부 정보를 사용하도록 장치를 다시 활성화할 수 있습니다.*

  ***--nodefroute** 옵션을 사용하여 장치가 기본 경로를 사용하지 못하도록 합니다.*

- ***--no-activate -** 설치 환경에서 이 장치를 활성화하지 마십시오.*

  *기본적으로 **Anaconda**는 **--activate** 옵션에 관계없이 **Kickstart** 파일의 첫 번째 네트워크 장치를 활성화합니다. **no-activate** 옵션을 사용하여 기본 설정을 비활성화할 수 있습니다.*

- ***--BOOTPROTO= - dhcp,bootp,ibft** 또는 **static** 중 하나입니다. 기본 옵션은 **dhcp; dhcp** 및 **bootp** 옵션은 동일하게 취급됩니다. 장치의 **ipv4** 구성을 비활성화하려면 **--noipv4** 옵션을 사용합니다.*

  참고

  *이 옵션은 장치의 **ipv4** 구성을 구성합니다. **ipv6** 구성의 경우 **--ipv6** 및 **--ipv6gateway** 옵션을 사용합니다.*

  ***DHCP** 방법은 **DHCP** 서버 시스템을 사용하여 네트워킹 구성을 가져옵니다. **BOOTP** 방법은 비슷합니다, 네트워크 구성을 공급하는 **BOOTP** 서버가 필요합니다. **DHCP**를 사용하도록 시스템을 실행하려면 다음을 수행합니다.*

```
network --bootproto=dhcp
```

BOOTP를 사용하도록 시스템을 지시하려면 **Kickstart** 파일에서 다음 행을 사용합니다.

```
network --bootproto=bootp
```

iBFT에서 지정된 구성을 사용하도록 머신을 지정하려면 다음을 사용합니다.

```
network --bootproto=ibft
```

정적 방법을 사용하려면 **Kickstart** 파일에서 최소한 **IP** 주소와 넷마스크를 지정해야 합니다. 이 정보는 정적이며 설치 중 및 설치 후 사용됩니다.

모든 정적 네트워킹 구성 정보를 한 줄에 지정해야 합니다. 명령줄에서 가능한 백슬래시(\)를 사용하여 행을 래핑할 수 없습니다.

```
network --bootproto=static --ip=10.0.2.15 --netmask=255.255.255.0 --gateway=10.0.2.254 --
nameserver=10.0.2.1
```

동시에 여러 이름 서버를 구성할 수도 있습니다. 이렇게 하려면 **--nameserver=** 옵션을 한 번 사용하고 각 **IP** 주소를 쉼표로 구분하여 지정합니다.

```
network --bootproto=static --ip=10.0.2.15 --netmask=255.255.255.0 --gateway=10.0.2.254 --
nameserver=192.168.2.1,192.168.3.1
```

- **--device= - network** 명령을 사용하여 구성할 장치(및 **Anaconda**에서 최종적으로 활성화)를 지정합니다.

  network 명령을 처음 사용할 때 **--device=** 옵션이 없는 경우, 사용 가능한 경우 **inst.ks.device= Anaconda** 부팅 옵션의 값이 사용됩니다. 이 동작은 더 이상 사용되지 않는 동작으로 간주됩니다. 대부분의 경우 항상 모든 **network** 명령에 **--device=** 를 지정해야 합니다.

중요

--device= 옵션이 누락된 경우 동일한 **Kickstart** 파일에서 후속 네트워크 명령의 동작이 지정되지 않습니다. 첫 번째 명령 이외의 네트워크 명령에 이 옵션을 지정해야 합니다.

다음 방법 중 하나로 활성화할 장치를 지정할 수 있습니다.

- 인터페이스의 장치 이름(예: **em1**)
- 인터페이스의 **MAC** 주소(예: **01:23:45:67:89:ab**)
- **up** 상태의 링크를 사용하여 첫 번째 인터페이스를 지정하는 키워드 링크
- **pxelinux**가 **BOOTIF** 변수에서 설정한 **MAC** 주소를 사용하는 키워드 **bootif**. **pxelinux.cfg** 파일에서 **IPAPPEND 2** 를 설정하여 **pxelinux** 변수를 **BOOTIF** 변수를 설정하도록 설정합니다.

예를 들어 다음과 같습니다.

```
network --bootproto=dhcp --device=em1
```

- **--IP= -** 장치의 **IP** 주소입니다.
- **--ipv6= -** 장치의 **IPv6** 주소 형식 **[/**접두사 길이**] -** 예: **3ffe:ffff:0:1::1/128.** 접두사 가 생략되면 **64** 가 사용됩니다. 자동 구성에 **auto** 를 사용하거나 **DHCPv6** 전용 구성에 **dhcp** 를 사용할 수도 있습니다(라우터 알림 없음).
- **--gateway= -** 기본 게이트웨이를 단일 **IPv4** 주소로 사용합니다.
- **--ipv6gateway= -** 기본 게이트웨이를 단일 **IPv6** 주소로 사용합니다.
-

--nodefroute - 기본 경로로 설정된 인터페이스를 해제합니다. 예를 들어 iSCSI 대상의 별도의 서브넷에 NIC가 있는 경우 --activate= 옵션을 사용하여 추가 장치를 활성화하면 이 옵션을 사용합니다.

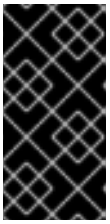
- *--nameserver= - DNS 이름 서버, IP 주소로. 둘 이상의 이름 서버를 지정하려면 이 옵션을 한 번 사용하고 각 IP 주소를 쉼표로 구분합니다.*

- *--netmask= - 설치된 시스템의 네트워크 마스크입니다.*

- *--hostname= - 대상 시스템의 호스트 이름을 구성하는 데 사용됩니다. 호스트 이름은 hostname.domainname 형식의 FQDN(정규화된 도메인 이름)이거나 도메인이 없는 짧은 호스트 이름일 수 있습니다. DHCP(Dynamic Host Configuration Protocol) 서비스를 사용하여 연결된 시스템에 도메인 이름을 자동으로 할당하는 경우 짧은 호스트 이름만 지정합니다.*

  대상 시스템의 호스트 이름만 구성하려면 network 명령에 --hostname 옵션을 사용하고 다른 옵션은 포함하지 마십시오.

  호스트 이름을 구성할 때 추가 옵션을 제공하는 경우 network 명령은 지정된 옵션을 사용하여 장치를 구성합니다. --device 옵션을 사용하여 구성할 장치를 지정하지 않으면 default --device 링크 값이 사용됩니다. 또한 --bootproto 옵션을 사용하여 프로토콜을 지정하지 않으면 장치가 기본적으로 DHCP를 사용하도록 구성됩니다.



중요

네트워크에서 DHCP 서비스를 제공하지 않는 경우 항상 FQDN을 시스템의 호스트 이름으로 사용하십시오.

- *--ethtool= - ethtool 프로그램으로 전달될 네트워크 장치에 대해 추가로 하위 수준 설정을 지정합니다.*

- *--ONBOOT= - 부팅 시 장치를 활성화할지 여부입니다.*

- *--dhcpclass= - DHCP 클래스.*

- *--MTU= - 장치의 MTU입니다.*

- ***--noipv4*** *- 이 장치에서* ***IPv4****를 비활성화합니다.*

- ***--noipv6*** *- 이 장치에서* ***IPv6****를 비활성화합니다.*

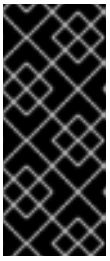
- ***--bondslaves=*** *- 이 옵션을 사용하면* ***--bondslaves=*** *옵션에 정의된 보조 장치를 사용하여* ***--device=*** *옵션으로 지정된 본딩 장치가 생성됩니다. 예를 들어 다음과 같습니다.*

```
network --device=bond0 --bondslaves=em1,em2
```

위의 명령은 ***em1*** *및* ***em2*** *인터페이스를 보조 장치로 사용하여* ***bond0*** *이라는 본딩 장치를 생성합니다.*

- ***--bondopts= - --bondslaves=*** *및* ***--device=*** *옵션을 사용하여 지정된 본딩 인터페이스에 대한 선택적 매개변수 목록입니다. 이 목록의 옵션은 쉼표(“,”) 또는 세미콜론(;)으로 구분되어야 합니다.“ ” 옵션 자체에 쉼표가 포함된 경우* ***Semicolon****을 사용하여 옵션을 구분합니다. 예를 들어 다음과 같습니다.*

```
network --bondopts=mode=active-backup,balance-rr;primary=eth1
```



중요

--bondopts=mode= *매개변수는* ***0*** *또는* ***3*** *과 같은 숫자 표현이 아닌* ***balance-rr*** *또는* ***broadcast*** *와 같은 전체 모드 이름만 지원합니다. 사용 가능하고 지원되는 모드 목록은 네트워킹 가이드 구성 및 관리를 참조하십시오.*

- ***--vlanid= - - --device=*** *에 지정된 장치를 상위로 사용하여 생성된 장치의* ***VLAN(****가상* ***LAN)*** ***ID*** *번호(****802.1q*** *태그)를 지정합니다. 예를 들어 네트워크* ***--device=em1 --vlanid=171*** *은 가상* ***LAN*** *장치* ***em1.171*** *을 생성합니다.*

- ***--interfaceName= -*** *가상* ***LAN*** *장치의 사용자 지정 인터페이스 이름을 지정합니다. 이 옵션은* ***--vlanid=*** *옵션으로 생성한 기본 이름이 바람직하지 않은 경우 사용해야 합니다. 이 옵션은* ***--vlanid=*** *과 함께 사용해야 합니다. 예를 들어 다음과 같습니다.*

```
network --device=em1 --vlanid=171 --interfacename=vlan171
```

위의 명령은 ***ID****가* ***171*** *인* ***em1*** *장치에* ***vlan171*** *이라는 가상* ***LAN*** *인터페이스를 생성합니다.*

*인터페이스 이름은 임의의(예: **my-vlan**)일 수 있지만 특정 경우에는 다음 규칙을 따라야 합니다.*

- *이름에 점(**..**)이 포함된 경우 **NAME.ID** 형식을 사용해야 합니다. **NAME** 은 임의의이지만 **ID** 는 **VLAN ID**여야 합니다. 예: **em1.171** 또는 **my-vlan.171.***
- ***vlan** 으로 시작하는 이름은 **vlanID** 형식이어야 합니다(예: **vlan171**).*

- ***--teamslaves= - --device=** 옵션으로 지정된 팀 장치는 이 옵션에 지정된 보조 장치를 사용하여 생성됩니다. 보조 장치는 쉼표로 구분됩니다. 보조 장치 뒤에는 \ 문자로 이스케이프된 작은따옴표가 있는 단일 인용된 **JSON** 문자열인 구성이 올 수 있습니다. 예를 들어 다음과 같습니다.*

```
network --teamslaves="p3p1'{\"prio\": -10, \"sticky\": true}',p3p2'{\"prio\": 100}'"
```

*또한 **--teamconfig=** 옵션도 참조하십시오.*

- ***--teamconfig= - double-quoted** 팀 장치 구성: \ 문자로 큰따옴표로 이스케이프된 **JSON** 문자열입니다. 장치 이름은 **--device=** 옵션과 해당 보조 장치 및 **--teamslaves=** 옵션에 의해 지정됩니다. 예를 들어 다음과 같습니다.*

```
network --device team0 --activate --bootproto static --ip=10.34.102.222 --
netmask=255.255.255.0 --gateway=10.34.102.254 --nameserver=10.34.39.2 --
teamslaves="p3p1'{\"prio\": -10, \"sticky\": true}',p3p2'{\"prio\": 100}'" --teamconfig="
{\"runner\": {\"name\": \"activebackup\"}}"
```

- ***--bridgeslaves= -** 이 옵션을 사용하면 **--device=** 옵션을 사용하여 지정된 장치 이름으로 네트워크 브리지가 생성되고 **--bridgeslaves=** 옵션에 정의된 장치가 브리지에 추가됩니다. 예를 들어 다음과 같습니다.*

```
network --device=bridge0 --bridgeslaves=em1
```

- ***--bridgeopts= -** 콤마로 구분된 브릿지 인터페이스의 매개변수 목록입니다. 사용 가능한 값은 **stp,priority,forward-delay,hello-time,max-age, ageing-time** 입니다. 이러한 매개변수에 대한 자세한 내용은 **nm-settings(5)** 도움말 페이지의 브리지 설정 테이블 또는 네트워크 구성 설정 사양 을 참조하십시오.*

네트워크 브리징에 대한 일반적인 정보는 네트워킹 구성 및 관리 문서를 참조하십시오.

- ***--bindto=mac*** *- 설치된 시스템의 장치 구성(**ifcfg**) 파일을 기본 바인딩 인터페이스 이름 (**DEVICE**) 대신 장치 **MAC** 주소(**HWADDR**)에 바인딩합니다. 이 옵션은* ***--device=*** *옵션과 독립적입니다 -* ***--bindto=mac*** *는 동일한 네트워크 명령이 장치 이름, 링크 또는* ***bootif*** *를 지정하는 경우에도 적용됩니다.*

참고

- ***eth0*** *과 같은* ***ethN*** *장치 이름은 이름 지정 체계의 변경으로 인해* ***Red Hat Enterprise Linux 8***에서 더 이상 사용할 수 없습니다. 장치 이름 지정 체계에 대한 자세한 내용은 업스트림 문서 ***Predictable Network Interface Names*** 를 참조하십시오.

- ***Kickstart*** *옵션 또는 부팅 옵션을 사용하여 네트워크에서 설치 리포지토리를 지정했지만 설치 시작 시 사용할 수 있는 네트워크가 없는 경우 설치 프로그램은 설치 요약 창을 표시하기 전에 네트워크 연결을 설정하는 네트워크 구성 창이 표시됩니다. 자세한 내용은 표준* ***RHEL 8*** *설치 문서의 네트워크 및 호스트 이름 옵션 구성 섹션을 참조하십시오.*

B.4.2. 영역

realm Kickstart *명령은 선택 사항입니다. 이를 사용하여* ***Active Directory*** *또는* ***IPA*** *도메인에 연결합니다. 이 명령에 대한 자세한 내용은* ***realm(8)*** *매뉴얼 페이지의 조인 섹션을 참조하십시오.*

구문

```
realm join [OPTIONS] domain
```

필수 옵션

- ***domain*** *- 조인할 도메인입니다.*

옵션

- ***-- computer-ou=OU=*** *- 컴퓨터 계정을 생성하기 위해 조직 단위의 고유 이름을 제공합니다. 고유 이름의 정확한 형식은 클라이언트 소프트웨어 및 멤버십 소프트웨어에 따라 다릅니다. 고유 이름의 루트* ***DSE*** *부분은 일반적으로 생략될 수 있습니다.*

- ***--no-password -*** *암호 없이 자동으로 참여합니다.*

- ***--one-time-password= -*** *일회성 암호를 사용하여 참여합니다. 모든 유형의 영역에서는 이 작업을 수행할 수 없습니다.*

- ***--client-software= -*** *이 클라이언트 소프트웨어를 실행할 수 있는 영역만 참여합니다. 유효한 값에는* ***sssd*** *및* ***winbind*** *가 포함됩니다. 모든 영역이 모든 값을 지원하는 것은 아닙니다. 기본적으로 클라이언트 소프트웨어는 자동으로 선택됩니다.*

- ***--server-software= -*** *이 서버 소프트웨어를 실행할 수 있는 영역만 참여합니다. 가능한 값에는* ***active-directory*** *또는* ***freeipa*** *가 포함됩니다.*

- ***--membership-software= -*** *영역에 가입할 때 이 소프트웨어를 사용합니다. 유효한 값에는* ***samba*** *및* ***adcli*** *가 포함됩니다. 모든 영역이 모든 값을 지원하는 것은 아닙니다. 기본적으로 멤버십 소프트웨어는 자동으로 선택됩니다.*

B.5. 스토리지 처리를 위한 KICKSTART 명령

이 섹션의 ***Kickstart*** *명령은 장치, 디스크, 파티션,* ***LVM*** *및 파일 시스템과 같은 스토리지 요소를 구성합니다.*

B.5.1. 장치(더 이상 사용되지 않음)

장치 ***Kickstart*** *명령은 선택 사항입니다. 추가 커널 모듈을 로드하려면 이 모듈을 사용합니다.*

대부분의 ***PCI*** *시스템에서 설치 프로그램은 이더넷 및* ***SCSI*** *카드를 자동으로 감지합니다. 그러나 이전 시스템과 일부* ***PCI*** *시스템에서는* ***Kickstart****에 적절한 장치를 찾으려면 힌트가 필요합니다. 설치 프로그램에 추가 모듈을 설치하도록 지시하는* ***device*** *명령은 다음 형식을 사용합니다.*

구문

```
device moduleName --opts=options
```

옵션

- **MODULE NAME -** 설치할 커널 모듈의 이름으로 바꿉니다.
- **--opts= -** 커널 모듈에 전달할 옵션 예를 들어 다음과 같습니다.

```
device --opts="aic152x=0x340 io=11"
```

B.5.2. autopart

autopart Kickstart 명령은 선택 사항입니다. 자동으로 파티션을 생성합니다.

자동으로 생성된 파티션은 루트(/) 파티션(**GiB** 또는 더 큰), 스왑 파티션 및 아키텍처에 적합한 **/boot** 파티션입니다. 충분히 큰 드라이브(**50GiB** 등)에서 이 드라이브는 **/home** 파티션도 생성합니다.

구문

```
autopart OPTIONS
```

옵션

- **--type= -** 사용할 사전 정의된 자동 파티션 체계 중 하나를 선택합니다. 다음 값을 허용합니다.
  - **LVM: LVM** 파티션 스키마.
  - **plain: LVM**이 없는 정규 파티션입니다.
  - **thinp: LVM** 씬 프로비저닝 파티션 스키마.

사용 가능한 파티션 체계에 대한 설명은 **C.1절. “지원되는 장치 유형”** 을 참조하십시오.

- *--fstype= - 사용 가능한 파일 시스템 유형 중 하나를 선택합니다. 사용 가능한 값은 ext2,ext3,ext4,xfs, vfat 입니다. 기본 파일 시스템은 xfs 입니다. 이러한 파일 시스템에 대한 자세한 내용은 C.2절. "지원되는 파일 시스템" 을 참조하십시오.*

- *--nohome - /home 파티션의 자동 생성을 비활성화합니다.*

- *--nolvm - 자동 파티셔닝을 위해 LVM을 사용하지 마십시오. 이 옵션은 --type=plain 과 같습니다.*

- *--noboot - /boot 파티션을 생성하지 않습니다.*

- *--noswap - 스왑 파티션을 생성하지 마십시오.*

- *--encrypted - Linux 통합 키 설정(LUKS)으로 모든 파티션을 암호화합니다. 이는 수동 그래픽 설치 중 초기 파티션 화면에서 파티션 암호화 확인란을 확인하는 것과 동일합니다.*

참고

하나 이상의 파티션을 암호화할 때 Anaconda는 256비트의 엔트로피를 수집하여 파티션을 안전하게 암호화하려고 합니다. 엔트로피 수집에는 약간의 시간이 걸릴 수 있습니다. 충분한 엔트로피가 수집되었는지에 관계없이 프로세스가 최대 10분 후에 중지됩니다.

이 프로세스는 설치 시스템과 상호 작용하여 정지할 수 있습니다(키보드에서 이동 또는 마우스 이동). 가상 머신에 설치하는 경우 virtio-rng 장치(가상 임의 번호 생성기)를 게스트에 연결할 수도 있습니다.

- *--LUKS-version=LUKS_VERSION - 파일 시스템을 암호화하는 데 사용할 LUKS 형식의 버전을 지정합니다. 이 옵션은 --encrypted가 지정된 경우에만 의미가 있습니다.*

- *--passphrase= - 암호화된 모든 장치에 기본 시스템 전체 암호를 제공합니다.*

- *--escrowcert= URL_of_X.509_certificate - 암호화된 모든 볼륨의 데이터 암호화 키를 /root 의 파일로 저장하고, URL_of_X.509_certificate로 지정된 URL에서 X.509 인증서를 사용하*

여 암호화됩니다. 키는 각 암호화된 볼륨에 대해 별도의 파일로 저장됩니다. 이 옵션은 **--encrypted**가 지정된 경우에만 의미가 있습니다.

- **--backuppassphrase** - 무작위로 생성된 각 볼륨에 암호를 추가합니다. 이러한 암호를 **/root**에 별도의 파일에 저장하고 **--escrowcert** 로 지정된 **X.509** 인증서를 사용하여 암호화됩니다. 이 옵션은 **--escrowcert**가 지정된 경우에만 의미가 있습니다.

- **--cipher=** - Anaconda 기본 **aes-xts-plain64** 가 만족스럽지 않은 경우 사용할 암호화 유형을 지정합니다. 이 옵션을 **--encrypted** 옵션과 함께 사용해야 합니다. 이 옵션 자체는 적용되지 않습니다. 사용 가능한 암호화 유형은 보안 강화 문서에 나열되어 있지만 **aes-xts-plain64** 또는 **aes-cbc-essiv:sha256** 을 사용하는 것이 좋습니다.

- **--PBKDF=PBKDF** - LUKS 키 lot에 대한 PBKDF(암호 기반 키 파생 기능) 알고리즘을 설정합니다. 도움말 페이지 **cryptsetup(8)** 도 참조하십시오. 이 옵션은 **--encrypted**가 지정된 경우에만 의미가 있습니다.

- **--PBKDF-memory=PBKDF_MEMORY** - PBKDF에 대한 메모리 비용을 설정합니다. 도움말 페이지 **cryptsetup(8)** 도 참조하십시오. 이 옵션은 **--encrypted**가 지정된 경우에만 의미가 있습니다.

- **--PBKDF-time=PBKDF_TIME** - PBKDF 암호 처리와 함께 사용할 밀리초 수를 설정합니다. **man** 페이지 **cryptsetup(8)** 에서도 **--iter-time** 을 참조하십시오. 이 옵션은 **--encrypted** 가 지정되고 **--pbkdf-iterations** 와 상호 배타적인 경우에만 의미가 있습니다.

- **--PBKDF-iterations=PBKDF_ITERATIONS** - 반복 횟수를 직접 설정하고 PBKDF 벤치마크를 방지합니다. **man** 페이지 **cryptsetup(8)** 에서 **--pbkdf-force-iterations** 도 참조하십시오. 이 옵션은 **--encrypted** 가 지정되고 **--pbkdf-time** 과 상호 배타적인 경우에만 의미가 있습니다.

참고

- **autopart** 옵션은 동일한 **Kickstart** 파일의 **part/partition,raid,logvol** 또는 **volgroup** 옵션과 함께 사용할 수 없습니다.

- **autopart** 명령은 필수는 아니지만 **Kickstart** 스크립트에 부분 또는 마운트 명령이 없는 경우 이를 포함해야 합니다.

- **CMS** 유형의 단일 **FBA DASD**에 설치할 때 **autopart --nohome Kickstart** 옵션을 사용하는 것이 좋습니다. 이렇게 하면 설치 프로그램이 별도의 **/home** 파티션을 생성하지 않습니다. 그런

다음 설치가 성공적으로 진행됩니다.

- ***LUKS** 암호를 분실하는 경우 암호화된 파티션과 해당 데이터에 완전히 액세스할 수 없습니다. 분실된 암호를 복구할 방법은 없습니다. 그러나 **--escrowcert**를 사용하여 암호화 암호를 저장하고 **--backuppassphrase** 옵션을 사용하여 백업 암호화 암호를 생성할 수 있습니다.*

- ***autopart ,autopart --type=lvm** 또는 **autopart =thinp** 를 사용할 때 디스크 섹터 크기가 일관되게 유지되도록 합니다.*

B.5.3. 부트로더(필수)

*부트로더 **Kickstart** 명령이 필요합니다. 부트 로더를 설치하는 방법을 지정합니다.*

구문

```
bootloader [OPTIONS]
```

옵션

- ***--append= -** 추가 커널 매개변수를 지정합니다. 여러 매개변수를 지정하려면 공백으로 구분합니다. 예를 들어 다음과 같습니다.*

  ```
  bootloader --location=mbr --append="hdd=ide-scsi ide=nodma"
  ```

  ***rhgb** 및 **quiet** 매개 변수는 **plymouth** 패키지가 설치될 때 자동으로 추가됩니다. 여기에서 지정하지 않거나 **--append=** 명령을 전혀 사용하지 않습니다. 이 동작을 비활성화하려면 **plymouth** 의 설치를 명시적으로 허용하지 않습니다.*

  ```
  %packages
  -plymouth
  %end
  ```

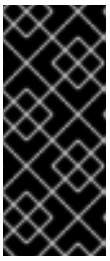
  *이 옵션은 최신 프로세서 **(CVE-2017-5754, CVE-2017-5753** 및 **CVE-2017-5753)**에서 발견된 **Meltdown** 및 **Spectre speculative** 실행 취약점을 완화하기 위해 구현된 메커니즘을 비활성화하는 데 유용합니다. 경우에 따라 이러한 메커니즘이 불필요할 수 있으며 이를 사용하도록 설정하면 보안이 향상되지 않고 성능이 저하됩니다. 이러한 메커니즘을 비활성화하려면 **AMD64/Intel 64***

시스템에서 **Kickstart** 파일(예: **bootloader --append="nopti noibrs noibpb "**)에 옵션을 추가하십시오.

주의

취약점 완화 메커니즘을 비활성화하기 전에 시스템이 공격 위험이 없는지 확인하십시오. **Meltdown** 및 **Spectre** 취약점에 대한 정보는 **Red Hat** 취약점 대응 문서 를 참조하십시오.

- **--boot-drive= -** 부트 로더가 작성되어야 하는 드라이브를 지정하므로 컴퓨터가 부팅될 드라이브를 지정합니다. 다중 경로 장치를 부팅 드라이브로 사용하는 경우 **disk/by-id/dm-uuid-mpath-WWID** 이름을 사용하여 장치를 지정합니다.



중요

--boot-drive= 옵션은 현재 **zipl** 부트 로더를 사용하는 **64**비트 **IBM Z** 시스템에 **Red Hat Enterprise Linux** 설치에서 무시되고 있습니다. **zipl** 이 설치되면 자체적으로 부팅 드라이브가 결정됩니다.

- **--leavebootorder -** 설치 프로그램은 **Red Hat Enterprise Linux 8**을 부트 로더에 설치된 시스템 목록 상단에 추가하고 기존 항목뿐만 아니라 모든 기존 항목을 보존합니다.

중요

이 옵션은 **Power** 시스템에만 적용되며 **UEFI** 시스템은 이 옵션을 사용해서는 안 됩니다.

- **--driveorder= - BIOS** 부팅 순서에서 첫 번째 드라이브를 지정합니다. 예를 들어 다음과 같습니다.

```
bootloader --driveorder=sda,hda
```

- **--location= -** 부트 레코드가 기록되는 위치를 지정합니다. 유효한 값은 다음과 같습니다.

- ***MBR -** 기본 옵션입니다. 드라이브가 마스터 부팅 레코드(**MBR**) 또는 **GUID** 파티션 테이블(**GPT**) 체계를 사용하는지에 따라 다릅니다.*

  ***GPT** 형식의 디스크에서 이 옵션은 부트 로더의 **1.5** 단계를 **BIOS** 부팅 파티션에 설치합니다.*

  ***MBR** 형식의 디스크에서 **1.5** 단계는 **MBR**과 첫 번째 파티션 사이의 빈 공간에 설치됩니다.*

- *파티션 **-** 커널이 포함된 파티션의 첫 번째 섹터에 부트 로더를 설치합니다.*

- ***none -** 부트 로더를 설치하지 않습니다.*

대부분의 경우 이 옵션을 지정할 필요가 없습니다.

- ***--nombr -** 부트 로더를 **MBR**에 설치하지 마십시오.*

- ***--password= - GRUB2**를 사용하는 경우 이 옵션으로 지정된 부트 로더 암호를 설정합니다. 이는 임의의 커널 옵션을 전달할 수 있는 **GRUB2** 쉘에 대한 액세스를 제한하는 데 사용해야 합니다.*

  *암호를 지정하면 **GRUB2**에서도 사용자 이름을 요청합니다. 사용자 이름은 항상 **root** 입니다.*

- ***--iscrypted -** 일반적으로 **--password=** 옵션을 사용하여 부트 로더 암호를 지정하면 일반 텍스트의 **Kickstart** 파일에 저장됩니다. 암호를 암호화하려면 이 옵션과 암호화된 암호를 사용합니다.*

  *암호화된 암호를 생성하려면 **grub2-mkpasswd-pbkdf2** 명령을 사용하고 사용하려는 암호를 입력하고 명령 출력(**grub.pbkdf2**로 시작하는 해시)을 **Kickstart** 파일에 복사합니다. 암호화된 암호가 있는 예제 **Kickstart** 항목은 다음과 유사합니다.*

```
bootloader --iscrypted --
password=grub.pbkdf2.sha512.10000.5520C6C9832F3AC3D149AC0B24BE69E2D4FB0DBE
EDBD29CA1D30A044DE2645C4C7A291E585D4DC43F8A4D82479F8B95CA4BA4381F8550
```

```
510B75E8E0BB2938990.C688B6F0EF935701FF9BD1A8EC7FE5BD2333799C98F28420C5
CC8F1A2A233DE22C83705BB614EA17F3FDFDF4AC2161CEA3384E56EB38A2E39102F53
34C47405E
```

- **--timeout= -** 기본 옵션을 부팅하기 전에 부트 로더가 대기하는 시간을 지정합니다(초 단위).

- **--default= -** 부트 로더 구성에서 기본 부트 이미지를 설정합니다.

- **--extlinux - GRUB2** 대신 **extlinux** 부트 로더를 사용합니다. 이 옵션은 **extlinux**에서 지원하는 시스템에서만 작동합니다.

- **--disabled -** 이 옵션은 더 강력한 버전의 **--location=none** 입니다. **--location=none** 은 부트 로더 설치를 비활성화하는 반면 **--disabled** 는 부트 로더 설치를 비활성화하고 부트 로더가 포함된 패키지 설치도 비활성화하여 공간을 절약합니다.

참고

- 모든 시스템에 부트 로더 암호를 설정하는 것이 좋습니다. 보호되지 않은 부트 로더를 사용하면 잠재적 공격자가 시스템의 부팅 옵션을 수정하고 시스템에 대한 무단 액세스를 얻을 수 있습니다.

- 경우에 따라 **AMD64, Intel 64** 및 **64**비트 **ARM** 시스템에 부트 로더를 설치하려면 특수 파티션이 필요합니다. 이 파티션의 유형과 크기는 부트 로더를 설치하는 디스크가 마스터 부팅 레코드 **(MBR)** 또는 **GUID** 파티션 테이블**(GPT)** 스키마를 사용하는지에 따라 달라집니다. 자세한 내용은 표준 **RHEL 8** 설치 문서의 부트 로더 구성 섹션을 참조하십시오.

- **sdX** (또는 **/dev/sdX**) 형식의 장치 이름은 재부팅 시 일관되게 유지되므로 일부 **Kickstart** 명령을 사용할 수 있습니다. 명령에서 장치 노드 이름을 호출할 때 **/dev/disk**의 모든 항목을 대신 사용할 수 있습니다. 예를 들면 다음과 같습니다.

```
part / --fstype=xfs --onpart=sda1
```

다음 중 하나와 유사한 항목을 사용할 수 있습니다.

```
part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
```

```
part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1
```

이렇게 하면 명령이 항상 동일한 스토리지 장치를 대상으로 합니다. 이는 특히 대규모 스토리지 환경에서 유용합니다. 스토리지 장치를 지속적으로 참조하기 위한 다양한 방법에 대한 자세한 내용은 스토리지 장치 관리 문서의 영구 이름 지정 장을 참조하십시오.

- **Red Hat Enterprise Linux 8**에서는 **--upgrade** 옵션이 더 이상 사용되지 않습니다.

B.5.4. zipl

zipl Kickstart 명령은 선택 사항입니다. **64**비트 **IBM Z**의 **ZIPL** 구성을 지정합니다.

옵션

- **--secure-boot -** 설치 시스템에서 지원하는 경우 보안 부팅을 활성화합니다.

참고

IBM z14 이후의 시스템에 설치하는 경우 설치된 시스템은 **IBM z14** 또는 이전 모델에서 부팅할 수 없습니다.

- **--force-secure-boot -** 무조건 보안 부팅을 활성화합니다.

참고

IBM z14 및 이전 모델에서는 설치가 지원되지 않습니다.

- **--no-secure-boot -** 보안 부팅을 비활성화합니다.

참고

Secure Boot는 **IBM z14** 및 이전 모델에서는 지원되지 않습니다. **IBM z14** 및 이전 모델에서 설치된 시스템을 부팅하려는 경우 **--no-secure-boot** 를 사용합니다.

B.5.5. clearpart

clearpart Kickstart 명령은 선택 사항입니다. 새 파티션을 만들기 전에 시스템에서 파티션을 제거합

니다. 기본적으로 파티션은 제거되지 않습니다.

구문

```
clearpart OPTIONS
```

옵션

- ***--all -** 시스템에서 모든 파티션을 지웁니다.*

  이 옵션은 연결된 네트워크 스토리지를 포함하여 설치 프로그램에서 연결할 수 있는 모든 디스크를 지웁니다. 이 옵션을 주의해서 사용하십시오.

  ***--drives=** 옵션을 사용하여 보존할 스토리지를 지우지 않고 나중에 네트워크 스토리지(예: **Kickstart** 파일의 **%post** 섹션)를 연결하거나 네트워크 스토리지에 액세스하는 데 사용되는 커널 모듈을 차단 목록에 추가하여 **--drives=** 옵션을 사용하여 보존할 스토리지를 방지할 수 있습니다.*

- ***--drives= -** 파티션을 지울 드라이브를 지정합니다. 예를 들어 다음은 기본 **IDE** 컨트롤러의 처음 두 드라이브의 모든 파티션을 지웁니다.*

  ```
  clearpart --drives=hda,hdb --all
  ```

  *다중 경로 장치를 지우려면 **disk/by-id/scsi-WWID** 형식을 사용합니다. 여기서 **WWID** 는 장치의 전역 식별자입니다. 예를 들어 **WWID 58095BEC5510947BE8C0360F604351918** 로 디스크를 지우려면 다음을 사용합니다.*

  ```
  clearpart --drives=disk/by-id/scsi-58095BEC5510947BE8C0360F604351918
  ```

  *이 형식은 모든 다중 경로 장치에 권장되지만 오류가 발생하면 **LVM(Logical Volume Management)**을 사용하지 않는 다중 경로 장치도 **Disk/by-id/dm-uuid-WWID** 형식으로 선택할 수 있습니다. 여기서 **WWID** 는 장치의 전역 식별자입니다. 예를 들어 **WWID 2416CD96995134CA5D787F00A5AA11017** 로 디스크를 지우려면 다음을 사용합니다.*

  ```
  clearpart --drives=disk/by-id/dm-uuid-mpath-2416CD96995134CA5D787F00A5AA11017
  ```

mpatha와 같은 장치 이름으로 다중 경로 장치를 지정하지 마십시오. 이와 같은 장치 이름은 특정 디스크에 한정되지 않습니다. 설치하는 동안 이름이 /dev/mpatha인 디스크가 예상했던 것과 다를 수 있습니다. 따라서 clearpart 명령은 잘못된 디스크를 대상으로 할 수 있습니다.

- *--initlabel - 포맷을 위해 지정된 각 아키텍처의 기본 디스크 레이블을 만들어 디스크(또는 디스크)를 초기화합니다(예: x86)의 msdos. --initlabel은 모든 디스크를 볼 수 있기 때문에 포맷할 드라이브만 연결되어 있는지 확인하는 것이 중요합니다.*

```
clearpart --initlabel --drives=names_of_disks
```

예를 들어 다음과 같습니다.

```
clearpart --initlabel --drives=dasda,dasdb,dasdc
```

- *--list= - 선택을 취소할 파티션을 지정합니다. 이 옵션은 사용된 경우 --all 및 --linux 옵션을 재정의합니다. 여러 드라이브에서 사용할 수 있습니다. 예를 들어 다음과 같습니다.*

```
clearpart --list=sda2,sda3,sdb1
```

- *--disklabel=LABEL - 기본 disklabel을 사용하도록 설정합니다. 플랫폼에 지원되는 디스크 레이블만 허용됩니다. 예를 들어 64비트 Intel 및 AMD 아키텍처에서는 msdos 및 gpt disklabels가 허용되지만 dasd 는 허용되지 않습니다.*

- *--Linux - 모든 Linux 파티션을 지웁니다.*

- *--none (기본값) - 파티션을 제거하지 않습니다.*

- *--CD L - 모든 LDL DASD를 CDL 형식으로 다시 포맷합니다.*

참고

- *sdX (또는 /dev/sdX) 형식의 장치 이름은 재부팅 시 일관되게 유지되므로 일부 Kickstart 명령을 사용할 수 있습니다. 명령에서 장치 노드 이름을 호출할 때 /dev/disk의 모든 항목을 대신 사용할 수 있습니다. 예를 들면 다음과 같습니다.*

```
part / --fstype=xfs --onpart=sda1
```

다음 중 하나와 유사한 항목을 사용할 수 있습니다.

```
part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
```

```
part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1
```

이렇게 하면 명령이 항상 동일한 스토리지 장치를 대상으로 합니다. 이는 특히 대규모 스토리지 환경에서 유용합니다. 스토리지 장치를 지속적으로 참조하기 위한 다양한 방법에 대한 자세한 내용은 스토리지 장치 관리 문서의 영구 이름 지정 장을 참조하십시오.

- **clearpart** 명령을 사용하면 논리 파티션에 **--onpart** 명령을 사용할 수 없습니다.

B.5.6. fcoe

fcoe Kickstart 명령은 선택 사항입니다. **ED(Enhanced Disk Drive Services)**에서 발견한 장치 외에도 자동으로 활성화해야 하는 **FCoE** 장치를 지정합니다.

구문

```
fcoe --nic=name [OPTIONS]
```

옵션

- **--NIC= (필수) -** 활성화할 장치의 이름입니다.
- **--dcB= - DCB(Data Center Bridging)** 설정을 구축합니다.
- **--autovlan - VLAN**을 자동으로 검색합니다. 이 옵션은 기본적으로 활성화되어 있습니다.

B.5.7. ignoredisk

ignoredisk Kickstart 명령은 선택 사항입니다. 이로 인해 설치 프로그램에서 지정된 디스크를 무시합

니다.

*이 기능은 자동 파티션을 사용하고 일부 디스크가 무시되는지 확인하려는 경우에 유용합니다. 예를 들어 **ignoredisk** 가 없는 경우 **Kickstart**가 실패하여 파티션 테이블이 반환되지 않는 **SAN**에 대한 수동 경로를 감지하므로 **Kickstart**가 실패할 수 있습니다.*

구문

```
ignoredisk --drives=drive1,drive2,... | --only-use=drive
```

옵션

- ***--drives=driveN,... - driveN** 을 **sda,sdb,..., hda ...** 등으로 바꿉니다.*
- ***--only-use=driveN,... -** 설치 프로그램에서 사용할 디스크 목록을 지정합니다. 다른 모든 디스크는 무시됩니다. 예를 들어 설치 중에 디스크 **sda**를 사용하고 다른 모든 디스크를 무시하려면 다음을 수행합니다.*

```
ignoredisk --only-use=sda
```

***LVM**을 사용하지 않는 다중 경로 장치를 포함하려면 다음을 수행합니다.*

```
ignoredisk --only-use=disk/by-id/dm-uuid-mpath-2416CD96995134CA5D787F00A5AA11017
```

***LVM**을 사용하는 다중 경로 장치를 포함하려면 다음을 수행합니다.*

```
ignoredisk --only-use==/dev/disk/by-id/dm-uuid-mpath-
```

```
bootloader --location=mbr
```

***--drives** 또는 **--only-use** 중 하나만 지정해야 합니다.*

참고

- ***Red Hat Enterprise Linux 8*에서는 *--interactive* 옵션이 더 이상 사용되지 않습니다. 이 옵션을 사용하면 사용자가 고급 스토리지 화면을 수동으로 탐색할 수 있습니다.**

- ***LVM(Logical Volume Management)*을 사용하지 않는 다중 경로 장치를 무시하려면 *disk/by-id/dm-uuid-mpath-WWID* 형식을 사용합니다. 여기서 *WWID* 는 장치의 전역 식별자입니다. 예를 들어 *WWID 2416CD96995134CA5D787F00A5AA11017* 인 디스크를 무시하려면 다음을 사용합니다.**

```
ignoredisk --drives=disk/by-id/dm-uuid-mpath-2416CD96995134CA5D787F00A5AA11017
```

- ***mpatha*와 같은 장치 이름으로 다중 경로 장치를 지정하지 마십시오. 이와 같은 장치 이름은 특정 디스크에 한정되지 않습니다. 설치하는 동안 이름이 */dev/mpatha*인 디스크가 예상했던 것과 다를 수 있습니다. 따라서 *clearpart* 명령은 잘못된 디스크를 대상으로 할 수 있습니다.**

- ***sdX* (또는 */dev/sdX*) 형식의 장치 이름은 재부팅 시 일관되게 유지되므로 일부 *Kickstart* 명령을 사용할 수 있습니다. 명령에서 장치 노드 이름을 호출할 때 */dev/disk*의 모든 항목을 대신 사용할 수 있습니다. 예를 들면 다음과 같습니다.**

```
part / --fstype=xfs --onpart=sda1
```

다음 중 하나와 유사한 항목을 사용할 수 있습니다.

```
part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
```

```
part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1
```

이렇게 하면 명령이 항상 동일한 스토리지 장치를 대상으로 합니다. 이는 특히 대규모 스토리지 환경에서 유용합니다. 스토리지 장치를 지속적으로 참조하기 위한 다양한 방법에 대한 자세한 내용은 스토리지 장치 관리 문서의 영구 이름 지정 장을 참조하십시오.

B.5.8. iscsi

***iscsi Kickstart* 명령은 선택 사항입니다. 설치하는 동안 연결할 추가 *iSCSI* 스토리지를 지정합니다.**

구문

```
iscsi --ipaddr=address [OPTIONS]
```

필수 옵션

- ***--ipaddr= (필수) - 연결할 대상의 IP 주소입니다.***

선택적 옵션

- ***--port= (필수) - 포트 번호입니다. 없는 경우 --port=3260이 기본적으로 자동으로 사용됩니다.***

- ***--target= - 대상 IQN(iSCSI 정규화된 이름)입니다.***

- ***--iface= - 기본적으로 네트워크 계층에 의해 결정된 인터페이스를 사용하는 대신 특정 네트워크 인터페이스에 연결을 바인딩합니다. 일단 사용되면 전체 Kickstart 파일에 있는 iscsi 명령의 모든 인스턴스에서 지정해야 합니다.***

- ***--user= - 대상 인증에 필요한 사용자 이름입니다.***

- ***--password= - 대상에 지정된 사용자 이름에 해당하는 암호입니다.***

- ***--reverse-user= - 역방향 CHAP 인증을 사용하는 대상에서 이니시에이터로 인증하는 데 필요한 사용자 이름입니다.***

- ***--reverse-password= - 개시자에 지정된 사용자 이름에 해당하는 암호입니다.***

참고

- ***iscsi 명령을 사용하는 경우 iscsiname 명령을 사용하여 iSCSI 노드에 이름을 할당해야 합니다. iscsiname 명령이 Kickstart 파일의 iscsi 명령보다 먼저 표시되어야 합니다.***

- ***가능한 경우 iscsi 명령을 사용하는 대신 시스템 BIOS 또는 펌웨어(Intel 시스템의 경우 iBFT)에서 iSCSI 스토리지를 구성합니다. Anaconda는 BIOS 또는 펌웨어에 구성된 디스크를 자동으로 감지하여 사용하며 Kickstart 파일에 특별한 구성이 필요하지 않습니다.***

- ***iscsi** 명령을 사용해야 하는 경우 설치가 시작될 때 네트워킹이 활성화되고 **clearpart** 또는 **ignoredisk** 와 같은 명령을 사용하여 **iSCSI** 디스크를 참조 하기 전에 **iscsi** 명령이 **Kickstart** 파일에 표시되는지 확인합니다.*

B.5.9. iscsiname

***iscsiname Kickstart** 명령은 선택 사항입니다. **iscsi** 명령으로 지정한 **iSCSI** 노드에 이름을 할당합니다.*

구문

```
iscsiname iqname
```

옵션

- ***iqname - iSCSI** 노드에 할당할 이름입니다.*

참고

- ***Kickstart** 파일에서 **iscsi** 명령을 사용하는 경우 **Kickstart** 파일에서 이전에 **iscsiname** 을 지정해야 합니다.*

B.5.10. logvol

***logvol Kickstart** 명령은 선택 사항입니다. **LVM(**논리 볼륨 관리**)**을 위한 논리 볼륨을 생성합니다.*

구문

```
logvol mntpoint --vgname=name --name=name [OPTIONS]
```

필수 옵션

mntpoint

파티션이 마운트된 마운트 지점입니다. 다음 양식 중 하나여야 합니다.

- ***/path***

  예를 들면 / 또는 /home입니다.

- ***swap***

  파티션은 스왑 공간으로 사용됩니다.

  *스왑 파티션의 크기를 자동으로 확인하려면 **--recommended** 옵션을 사용합니다.*

  ```
  swap --recommended
  ```

  *스왑 파티션의 크기를 자동으로 결정하고 시스템이 최대 절전할 수 있는 추가 공간을 허용하려면 **--hibernation** 옵션을 사용합니다.*

  ```
  swap --hibernation
  ```

  *할당된 크기는 **--recommended**로 할당된 스왑 공간과 시스템의 **RAM** 양과 동일합니다.*

  *이러한 명령으로 할당된 스왑 크기에 대해서는 **AMD64, Intel 64** 및 **64**비트 **ARM** 시스템의 권장 파티션 지정 스키마를 참조하십시오.*

--vgname=name

볼륨 그룹의 이름입니다.

--name=name

논리 볼륨의 이름입니다.

선택적 옵션

--noformat

기존 논리 볼륨을 사용하고 포맷하지 마십시오.

--useexisting

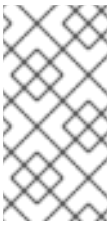
기존 논리 볼륨을 사용하고 다시 포맷하십시오.

--fstype=

*논리 볼륨의 파일 시스템 유형을 설정합니다. 유효한 값은 **xfs,ext2,ext3,ext4,swap, vfat** 입니다.*

--fsoptions=

*파일 시스템을 마운트할 때 사용할 옵션의 무료 양식 문자열을 지정합니다. 이 문자열은 설치된 시스템의 **/etc/fstab** 파일에 복사되며 따옴표로 묶어야 합니다.*



참고

***EFI** 시스템 파티션(**/boot/efi**)에서 **anaconda** 하드 코딩은 값을 코딩하고 지정된 **--fsoptions** 값을 무시합니다.*

--mkfsoptions=

*이 파티션의 파일 시스템을 만드는 프로그램에 전달할 추가 매개변수를 지정합니다. 인수 목록에서 처리가 수행되지 않으므로 **mkfs** 프로그램에 직접 전달할 수 있는 형식으로 제공해야 합니다. 즉, 파일 시스템에 따라 쉼표로 구분되거나 큰따옴표로 묶어야 합니다.*

--fsprofile=

*이 파티션의 파일 시스템을 만드는 프로그램에 전달할 사용 유형을 지정합니다. 사용 유형은 파일 시스템을 만들 때 사용할 다양한 튜닝 매개 변수를 정의합니다. 이 옵션을 사용하려면 파일 시스템에서 사용 유형의 개념을 지원해야 하며 유효한 유형을 나열하는 구성 파일이 있어야 합니다. **ext2,ext3** 및 **ext4** 의 경우 이 구성 파일은 **/etc/mke2fs.conf** 입니다.*

--label=

논리 볼륨의 레이블을 설정합니다.

--grow

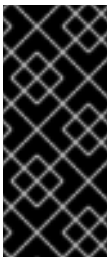
논리 볼륨을 확장하여 사용 가능한 공간(있는 경우) 또는 최대 크기(있는 경우)를 차지합니다. 옵션은 디스크 이미지에 최소 스토리지 공간을 미리 할당하고 볼륨을 늘리고 사용 가능한 공간을 차지하도록 하는 경우에만 사용해야 합니다. 물리적 환경에서 이것은 일회성 작업입니다. 그러나 가상 환경에서는 가상 머신이 가상 디스크에 데이터를 쓸 때 볼륨 크기가 다음과 같이 증가합니다.

--size=

*논리 볼륨의 크기(MiB)입니다. 이 옵션은 **--percent=** 옵션과 함께 사용할 수 없습니다.*

--percent=

*정적으로 크기의 논리 볼륨을 고려한 후 볼륨 그룹에서 사용 가능한 공간의 백분율로 논리 볼륨의 크기입니다. 이 옵션은 **--size=** 옵션과 함께 사용할 수 없습니다.*



중요

*새 논리 볼륨을 생성할 때 **--size=** 옵션을 사용하여 크기를 정적으로 지정하거나 **--percent=** 옵션을 사용하여 남은 여유 공간의 백분율로 지정해야 합니다. 동일한 논리 볼륨에서 이러한 옵션을 모두 사용할 수 없습니다.*

--maxsize=

*논리 볼륨이 확장되도록 설정된 경우 최대 크기(MiB)입니다. 여기에 **500**과 같은 정수 값을 지정합니다(단위를 포함하지 마십시오).*

--recommended

시스템 하드웨어에 따라 이 볼륨의 크기를 자동으로 결정하려면 논리 볼륨을 생성할 때 이 옵션을 사용합니다.

*권장 계획에 대한 자세한 내용은 **AMD64, Intel 64** 및 **64**비트 **ARM** 시스템의 권장 파티션 스키마를 참조하십시오.*

--resize

*논리 볼륨 크기 조정. 이 옵션을 사용하는 경우 **--useexisting** 및 **--size**도 지정해야 합니다.*

--encrypted

*이 논리 볼륨을 **--passphrase=** 옵션에 제공된 암호를 사용하여 **LUKS(Linux Unified Key Setup)**로 암호화하도록 지정합니다. 암호를 지정하지 않으면 설치 프로그램은 **autopart --passphrase** 명령으로 설정된 기본 시스템 전체 암호를 사용하거나 설치를 중지하고 기본값이 설정되지 않은 경우 암호를 제공하라는 메시지를 표시합니다.*

참고

*하나 이상의 파티션을 암호화할 때 **Anaconda**는 **256**비트의 엔트로피를 수집하여 파티션을 안전하게 암호화하려고 합니다. 엔트로피 수집에는 약간의 시간이 걸릴 수 있습니다. 충분한 엔트로피가 수집되었는지에 관계없이 프로세스가 최대 **10**분 후에 중지됩니다.*

*이 프로세스는 설치 시스템과 상호 작용하여 정지할 수 있습니다(키보드에서 이동 또는 마우스 이동). 가상 머신에 설치하는 경우 **virtio-rng** 장치(가상 임의 번호 생성기)를 게스트에 연결할 수도 있습니다.*

--passphrase=

*이 논리 볼륨을 암호화할 때 사용할 암호를 지정합니다. 이 옵션을 **--encrypted** 옵션과 함께 사용해야 합니다. 이 옵션 자체는 적용되지 않습니다.*

--cipher=

***Anaconda** 기본 **aes-xts-plain64** 가 만족스럽지 않은 경우 사용할 암호화 유형을 지정합니다. 이 옵션을 **--encrypted** 옵션과 함께 사용해야 합니다. 이 옵션 자체는 적용되지 않습니다. 사용 가능한 암호화 유형은 보안 강화 문서에 나열되어 있지만 **aes-xts-plain64** 또는 **aes-cbc-essiv:sha256** 을 사용하는 것이 좋습니다.*

--escrowcert=URL_of_X.509_certificate

*암호화된 모든 볼륨의 데이터 암호화 키를 **/root** 의 파일로 저장하고, **URL_of_X.509_certificate** 로 지정된 **URL**의 **X.509** 인증서를 사용하여 암호화됩니다. 키는 각 암호화된 볼륨에 대해 별도의 파일로 저장됩니다. 이 옵션은 **--encrypted**가 지정된 경우에만 의미가 있습니다.*

--luks-version=LUKS_VERSION

*파일 시스템을 암호화하는 데 사용할 **LUKS** 형식의 버전을 지정합니다. 이 옵션은 **--encrypted**가 지정된 경우에만 의미가 있습니다.*

--backuppassphrase

*암호화된 각 볼륨에 무작위로 생성된 암호를 추가합니다. 이러한 암호를 **/root** 에 별도의 파일에 저장하고 **--escrowcert** 로 지정된 **X.509** 인증서를 사용하여 암호화됩니다. 이 옵션은 **--escrowcert**가 지정된 경우에만 의미가 있습니다.*

--pbkdf=PBKDF

***LUKS** 키 **lot**에 대한 **PBKDF**(암호 기반 키 파생 기능) 알고리즘을 설정합니다. 도움말 페이지 **cryptsetup(8)** 도 참조하십시오. 이 옵션은 **--encrypted**가 지정된 경우에만 의미가 있습니다.*

--pbkdf-memory=PBKDF_MEMORY

PBKDF의 메모리 비용을 설정합니다. 도움말 페이지 cryptsetup(8) 도 참조하십시오. 이 옵션은 --encrypted가 지정된 경우에만 의미가 있습니다.

--pbkdf-time=PBKDF_TIME

PBKDF 암호 처리와 함께 사용할 시간(밀리초)을 설정합니다. man 페이지 cryptsetup(8) 에서도 --iter-time 을 참조하십시오. 이 옵션은 --encrypted 가 지정되고 --pbkdf-iterations 와 상호 배타적인 경우에만 의미가 있습니다.

--pbkdf-iterations=PBKDF_ITERATIONS

반복 횟수를 직접 설정하고 PBKDF 벤치마크를 방지합니다. man 페이지 cryptsetup(8) 에서 --pbkdf-force-iterations 도 참조하십시오. 이 옵션은 --encrypted 가 지정되고 --pbkdf-time 과 상호 배타적인 경우에만 의미가 있습니다.

--thinpool

씬 풀 논리 볼륨을 생성합니다. (마운트 지점 없음)

--metadatasize=size

새 씬 풀 장치의 메타데이터 영역 크기(MiB)를 지정합니다.

--chunksize=size

새 씬 풀 장치의 청크 크기(KiB)를 지정합니다.

--thin

thin 논리 볼륨을 만듭니다. (-poolname사용 필요)

--poolname=name

씬 논리 볼륨을 생성할 씬 풀의 이름을 지정합니다. --thin 옵션이 필요합니다.

--profile=name

씬 논리 볼륨에 사용할 구성 프로필 이름을 지정합니다. 사용하면 지정된 논리 볼륨의 메타데이터에도 이름이 포함됩니다. 기본적으로 사용 가능한 프로필은 default 및 thin-performance 이며 /etc/lvm/profile/ 디렉터리에 정의됩니다. 자세한 내용은 lvm(8) 도움말 페이지를 참조하십시오.

--cachepvs=

이 볼륨의 캐시로 사용해야 하는 특수한 물리 볼륨 목록입니다.

--cachemode=

이 논리 볼륨을 캐시하는 데 사용할 모드를 지정합니다(writeback 또는 writethrough).

참고

캐시된 논리 볼륨 및 모드에 대한 자세한 내용은 **lvmcache(7)** 도움말 페이지를 참조하십시오.

--cachesize=

MiB에 지정된 논리 볼륨에 연결된 캐시 크기입니다. 이 옵션에는 **--cachepvs=** 옵션이 필요합니다.

참고

- **Kickstart**를 사용하여 **Red Hat Enterprise Linux**를 설치할 때 논리 볼륨 및 볼륨 그룹 이름에 대시(**-**) 문자를 사용하지 마십시오. 이 문자를 사용하는 경우 설치가 정상적으로 완료되지만 **/dev/mapper/** 디렉터리에 모든 대시가 두 배가 되어 이러한 볼륨과 볼륨 그룹이 나열됩니다. 예를 들어 **logvol-01** 이라는 논리 볼륨이 포함된 **volgrp-01** 이라는 볼륨 그룹이 **/dev/mapper/volp-01-logvol-01** 로 나열됩니다.

  이 제한은 새로 생성된 논리 볼륨 및 볼륨 그룹 이름에만 적용됩니다. **--noformat** 옵션을 사용하여 기존 항목을 재사용하는 경우 해당 이름은 변경되지 않습니다.

- **LUKS** 암호를 분실하는 경우 암호화된 파티션과 해당 데이터에 완전히 액세스할 수 없습니다. 분실된 암호를 복구할 방법은 없습니다. 그러나 **--escrowcert**를 사용하여 암호화 암호를 저장하고 **--backuppassphrase** 옵션을 사용하여 백업 암호화 암호를 생성할 수 있습니다.

예제

- 먼저 파티션을 만들고 논리 볼륨 그룹을 만든 다음 논리 볼륨을 만듭니다.

```
part pv.01 --size 3000
volgroup myvg pv.01
logvol / --vgname=myvg --size=2000 --name=rootvol
```

- 먼저 파티션을 만들고 논리 볼륨 그룹을 만든 다음 볼륨 그룹의 나머지 공간 중 **90%**를 차지할 논리 볼륨을 만듭니다.

```
part pv.01 --size 1 --grow
volgroup myvg pv.01
logvol / --vgname=myvg --name=rootvol --percent=90
```

추가 리소스

- *논리 볼륨 구성 및 관리*

B.5.11. mount

***mount Kickstart** 명령은 선택 사항입니다. 기존 블록 장치에 마운트 지점을 할당하고 선택적으로 지정된 형식으로 다시 포맷합니다.*

구문

```
mount [OPTIONS] device mountpoint
```

필수 옵션:

- ***Device -** 마운트할 블록 장치입니다.*

- *마운트 지점 **-** 장치를 마운트할 위치 **. /** 또는 **/usr** 과 같은 유효한 마운트 지점이거나 장치를 마운트 해제할 수 있는 경우 없음 이어야 합니다(예: **swap**).*

선택적 옵션:

- ***--reformat= -** 장치를 다시 포맷해야 하는 새로운 형식(예: **ext4**)을 지정합니다.*

- ***--mkfsoptions= - --reformat=**에 지정된 새 파일 시스템을 생성하는 명령에 전달할 추가 옵션을 지정합니다. 여기에서 제공되는 옵션 목록은 처리되지 않으므로 **mkfs** 프로그램에 직접 전달할 수 있는 형식으로 지정해야 합니다. 옵션 목록은 파일 시스템에 따라 쉼표로 구분되거나 큰따옴표로 묶어야 합니다. 자세한 내용은 만들 파일 시스템의 **mkfs man** 페이지 (예: **mkfs.ext4(8)** 또는 **mkfs.xfs(8))**를 참조하십시오.*

- ***--mountoptions= -** 파일 시스템을 마운트할 때 사용할 옵션이 포함된 무료 양식 문자열을 지정합니다. 문자열은 설치된 시스템의 **/etc/fstab** 파일에 복사되며 큰따옴표로 묶어야 합니다. 기본 사항은 **mount(8)** 도움말 페이지에서 전체 마운트 옵션 및 **fstab(5)**를 참조하십시오.*

참고

- ***Kickstart*의 다른 스토리지 구성 명령과 달리 *mount*는 *Kickstart* 파일의 전체 스토리지 구성을 설명할 필요가 없습니다. 설명된 블록 장치가 시스템에 있는지 확인하기만 하면 됩니다. 그러나 마운트된 모든 장치로 스토리지 스택을 만들려면 *part* 와 같은 다른 명령을 사용하여 수행해야 합니다.**

- ***mount* 는 *part* ,*logvol*, 또는 동일한 *Kickstart* 파일에서 *auto part* 와 같은 다른 스토리지 관련 명령과 함께 사용할 수 없습니다.**

B.5.12. nvdimm

***nvdimm Kickstart* 명령은 선택 사항입니다. *NVDIMM(Non-Volatile Dual In-line Memory Module)* 장치에서 작업을 수행합니다.**

구문

```
nvdimm action [OPTIONS]
```

작업

- ***재구성 - 특정 NVDIMM* 장치를 지정된 모드로 재구성합니다. 또한 지정된 장치는 암시적으로 사용되도록 표시되어 있으므로 동일한 장치에 대한 후속 *nvdimm use* 명령이 중복됩니다. 이 작업은 다음 형식을 사용합니다.**

  ```
  nvdimm reconfigure [--namespace=NAMESPACE] [--mode=MODE] [--
  sectorsize=SECTORSIZE]
  ```

  - ***--namespace= -* 네임스페이스별 장치 사양입니다. 예를 들어 다음과 같습니다.**

    ```
    nvdimm reconfigure --namespace=namespace0.0 --mode=sector --sectorsize=512
    ```

  - ***--mode= -* 모드 사양입니다. 현재는 *Value* 섹터 만 사용할 수 있습니다.**

  - ***--sectorsize= -* 섹터 모드를 위한 섹터의 크기. 예를 들어 다음과 같습니다.**

```
nvdimm reconfigure --namespace=namespace0.0 --mode=sector --sectorsize=512
```

지원되는 섹터 크기는 **512** 및 **4096**바이트입니다.

- **use - NVDIMM** 장치를 설치 대상으로 지정합니다. 장치는 **nvdimm reconfigure** 명령을 통해 섹터 모드로 이미 구성되어 있어야 합니다. 이 작업은 다음 형식을 사용합니다.

```
nvdimm use [--namespace=NAMESPACE|--blockdevs=DEVICES]
```

  - **--namespace= -** 네임스페이스별로 장치를 지정합니다. 예를 들어 다음과 같습니다.

```
nvdimm use --namespace=namespace0.0
```

  - **--blockdevs= - NVDIMM** 장치에 해당하는 쉼표로 구분된 블록 장치를 지정합니다. 별표 * 와일드카드가 지원됩니다. 예를 들어 다음과 같습니다.

```
nvdimm use --blockdevs=pmem0s,pmem1s
nvdimm use --blockdevs=pmem*
```

참고

- 기본적으로 모든 **NVDIMM** 장치는 설치 프로그램에서 무시합니다. **nvdimm** 명령을 사용하여 이러한 장치에 설치를 활성화해야 합니다.

B.5.13. 부분 또는 파티션

일부 또는 파티션 **Kickstart** 명령이 필요합니다. 시스템에 파티션을 만듭니다.

구문

```
part|partition mntpoint --name=name --device=device --rule=rule [OPTIONS]
```

옵션

- ***mntpoint* -** *파티션이 마운트된 위치입니다. 값은 다음 형식 중 하나여야 합니다.*

  - ***/path***

    예를 들면 /, /usr,/home

  - ***swap***

    파티션은 스왑 공간으로 사용됩니다.

    스왑 파티션의 크기를 자동으로 확인하려면 ***--recommended*** *옵션을 사용합니다.*

    ```
    swap --recommended
    ```

    할당된 크기는 효율적이지만 시스템에 대해 정확하게 교정하지는 않습니다.

    스왑 파티션의 크기를 자동으로 결정하되 시스템의 추가 공간도 허용하려면 ***--hibernation*** *옵션을 사용합니다.*

    ```
    swap --hibernation
    ```

    할당된 크기는 ***--recommended****로 할당된 스왑 공간과 시스템의* ***RAM*** *양과 동일합니다.*

    이러한 명령으로 할당된 스왑 크기에 대해서는 ***AMD64, Intel 64*** *및* ***64****비트* ***ARM*** *시스템의* ***C.4****절. “권장 파티션 스키마”* *을 참조하십시오.*

  - ***RAID.id***

    파티션은 소프트웨어 ***RAID****에 사용됩니다(라이드 참조).*

  - ***pv.id***

파티션은 LVM에 사용됩니다(Log vol참조).

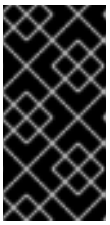
- ***biosboot***

  파티션은 BIOS 부팅 파티션에 사용됩니다. GUID 파티션 테이블(GPT)을 사용하는 BIOS 기반 AMD64 및 Intel 64 시스템에 1MiB BIOS 부팅 파티션이 필요합니다. 부트 로더가 여기에 설치됩니다. UEFI 시스템에서는 필요하지 않습니다. bootloader 명령도 참조하십시오.

- ***/boot/efi***

  EFI 시스템 파티션. UEFI 기반 AMD64, Intel 64 및 64비트 ARM에 50MiB EFI 파티션이 필요합니다. 권장 크기는 200MiB입니다. BIOS 시스템에서는 필요하지 않습니다. bootloader 명령도 참조하십시오.

- ***--size= -** 최소 파티션 크기(MiB)입니다. 여기에 500과 같은 정수 값을 지정합니다(단위를 포함하지 마십시오).*



중요

--size 값이 너무 작으면 설치에 실패합니다. --size 값을 필요한 최소 공간으로 설정합니다. 크기 권장 사항은 C.4절. "권장 파티션 스키마" 에서 참조하십시오.

- ***--grow -** 사용 가능한 공간을 채우기 위해 확장 가능한 공간(있는 경우) 또는 최대 크기 설정까지 해당 파티션을 지정합니다.*

참고

스왑 파티션에 --maxsize= 을 설정하지 않고 --grow= 를 사용하는 경우 Anaconda는 스왑 파티션의 최대 크기를 제한합니다. 실제 메모리가 2GiB 미만인 시스템의 경우 지정된 제한은 실제 메모리 양에 두 배입니다. 2GiB가 넘는 시스템의 경우 적용된 제한은 물리 메모리 크기와 2GiB의 크기입니다.

- ***--MaxSize= -** 파티션이 확장되도록 설정된 경우 최대 파티션 크기(MiB)입니다. 여기에 500과 같은 정수 값을 지정합니다(단위를 포함하지 마십시오).*

- ***--noformat - --onpart** 명령과 함께 사용하려면 파티션을 포맷하지 않도록 지정합니다.*

- ***--onpart=** 또는 **--usepart= -** 파티션을 배치할 장치를 지정합니다. 기존의 빈 장치를 사용하여 새로운 지정된 유형으로 포맷합니다. 예를 들어 다음과 같습니다.*

```
partition /home --onpart=hda1
```

***/dev/hda1** 에 **/home** 을 되게 합니다.*

이러한 옵션은 논리 볼륨에 파티션을 추가할 수도 있습니다. 예를 들어 다음과 같습니다.

```
partition pv.1 --onpart=hda2
```

*장치가 이미 시스템에 있어야 합니다 . **--onpart** 옵션은 생성되지 않습니다.*

*파티션이 아닌 전체 드라이브를 지정할 수도 있습니다. 이 경우 **Anaconda**는 파티션 테이블을 만들지 않고 드라이브를 포맷하고 사용할 수 있습니다. 그러나 이러한 방식으로 포맷된 장치에서 **GRUB2**의 설치는 지원되지 않으며 파티션 테이블이 있는 드라이브에 배치해야 합니다.*

```
partition pv.1 --onpart=hdb
```

- ***--ondisk=** 또는 **--ondrive= -** 기존 디스크에 파티션(**part** 명령을 통해 지정)을 만듭니다. 이 명령은 항상 파티션을 생성합니다. 특정 디스크에 파티션을 강제로 만듭니다. 예를 들어, **--ondisk=sdb**는 시스템의 두 번째 **SCSI** 디스크에 파티션을 둡니다.*

*LVM(Logical Volume Management)을 사용하지 않는 다중 경로 장치를 지정하려면 **disk/by-id/dm-uuid-mpath-WWID** 형식을 사용합니다. 여기서 **WWID** 는 장치의 전역 식별자입니다. 예를 들어 **WWID 2416CD96995134CA5D787F00A5AA11017** 디스크를 지정하려면 다음을 사용합니다.*

```
part / --fstype=xfs --grow --asprimary --size=8192 --ondisk=disk/by-id/dm-uuid-mpath-
2416CD96995134CA5D787F00A5AA11017
```

주의

mpatha와 같은 장치 이름으로 다중 경로 장치를 지정하지 마십시오. 이와 같은 장치 이름은 특정 디스크에 한정되지 않습니다. 설치하는 동안 이름이 /dev/mpatha인 디스크가 예상했던 것과 다를 수 있습니다. 따라서 part 명령은 잘못된 디스크를 대상으로 할 수 있습니다.

- ***--asprimary -** 파티션을 기본 파티션으로 할당합니다. 파티션을 주로 할당할 수 없는 경우 (일반적으로 너무 많은 기본 파티션이 할당되므로) 파티션 프로세스가 실패합니다. 이 옵션은 디스크가 **MBR(Master Boot Record)**을 사용하는 경우에만 의미가 있습니다. **GUID** 파티션 테이블 **(GPT)**의 경우 이 옵션에는 의미가 없습니다.*

- ***--fsprofile= -** 이 파티션에서 파일 시스템을 만드는 프로그램에 전달할 사용 유형을 지정합니다. 사용 유형은 파일 시스템을 만들 때 사용할 다양한 튜닝 매개 변수를 정의합니다. 이 옵션을 사용하려면 파일 시스템에서 사용 유형의 개념을 지원해야 하며 유효한 유형을 나열하는 구성 파일이 있어야 합니다. **ext2,ext3,ext4** 의 경우 이 구성 파일은 **/etc/mke2fs.conf** 입니다.*

- ***--mkfsoptions= -** 이 파티션에서 파일 시스템을 만드는 프로그램에 전달할 추가 매개변수를 지정합니다. 이는 **--fsprofile** 과 유사하지만 프로필 개념을 지원하는 시스템뿐만 아니라 모든 파일 시스템에서 작동합니다. 인수 목록에서 처리가 수행되지 않으므로 **mkfs** 프로그램에 직접 전달할 수 있는 형식으로 제공해야 합니다. 즉, 파일 시스템에 따라 쉼표로 구분되거나 큰따옴표로 묶어야 합니다.*

- ***--fstype= -** 파티션의 파일 시스템 유형을 설정합니다. 유효한 값은 **xfs,ext2,ext3,ext4,swap,vfat,efi** 및 **biosboot** 입니다.*

- ***--fsoptions -** 파일 시스템을 마운트할 때 사용할 옵션의 무료 양식 문자열을 지정합니다. 이 문자열은 설치된 시스템의 **/etc/fstab** 파일에 복사되며 따옴표로 묶어야 합니다.*

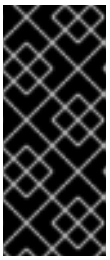
참고

***EFI** 시스템 파티션(**/boot/efi**)에서 **anaconda** 하드 코딩은 값을 코딩하고 지정된 **--fsoptions** 값을 무시합니다.*

- ***--label= -** 개별 파티션에 레이블을 할당합니다.*

- **--recommended** - 파티션의 크기를 자동으로 결정합니다.

  권장 체계에 대한 자세한 내용은 **AMD64, Intel 64** 및 **64**비트 **ARM**의 C.4절. “권장 파티션 스키마” 를 참조하십시오.



  중요

  이 옵션은 **/boot** 파티션 및 스왑 공간과 같은 파일 시스템을 만드는 파티션에만 사용할 수 있습니다. **LVM** 물리 볼륨 또는 **RAID** 멤버를 생성하는 데 사용할 수 없습니다.

- **--onbiosdisk** - **BIOS**에서 검색한 대로 특정 디스크에 파티션을 만듭니다.

- **--encrypted** - 이 파티션을 **--passphrase** 옵션에 제공된 암호를 사용하여 **LUKS(Linux Unified Key Setup)**로 암호화하도록 지정합니다. 암호를 지정하지 않으면 **Anaconda**에서 **autopart --passphrase** 명령으로 설정된 기본 시스템 전체 암호를 사용하거나 설치를 중지하고 기본값이 설정되지 않은 경우 암호를 입력하라는 메시지가 표시됩니다.

  참고

  하나 이상의 파티션을 암호화할 때 **Anaconda**는 **256**비트의 엔트로피를 수집하여 파티션을 안전하게 암호화하려고 합니다. 엔트로피 수집에는 약간의 시간이 걸릴 수 있습니다. 충분한 엔트로피가 수집되었는지에 관계없이 프로세스가 최대 **10**분 후에 중지됩니다.

  이 프로세스는 설치 시스템과 상호 작용하여 정지할 수 있습니다(키보드에서 이동 또는 마우스 이동). 가상 머신에 설치하는 경우 **virtio-rng** 장치(가상 임의 번호 생성기)를 게스트에 연결할 수도 있습니다.

- **--LUKS-version=LUKS_VERSION** - 파일 시스템을 암호화하는 데 사용할 **LUKS** 형식의 버전을 지정합니다. 이 옵션은 **--encrypted**가 지정된 경우에만 의미가 있습니다.

- **--passphrase=** - 이 파티션을 암호화할 때 사용할 암호를 지정합니다. 이 옵션을 **--encrypted** 옵션과 함께 사용해야 합니다. 이 옵션 자체는 적용되지 않습니다.

- **--cipher=** - **Anaconda** 기본 **aes-xts-plain64** 가 만족스럽지 않은 경우 사용할 암호화 유형을 지정합니다. 이 옵션을 **--encrypted** 옵션과 함께 사용해야 합니다. 이 옵션 자체는 적용되지 않

*습니다. 사용 가능한 암호화 유형은 보안 강화 문서에 나열되어 있지만 **aes-xts-plain64** 또는 **aes-cbc-essiv:sha256** 을 사용하는 것이 좋습니다.*

- *--escrowcert= URL_of_X.509_certificate - /root 에 있는 파일로 모든 암호화된 파티션의 데이터 암호화 키를 저장하고, URL_of_X.509_certificate로 지정된 URL에서 X.509 인증서를 사용하여 암호화됩니다. 키는 암호화된 각 파티션에 대해 별도의 파일로 저장됩니다. 이 옵션은 --encrypted가 지정된 경우에만 의미가 있습니다.*

- *--backuppassphrase - 임의로 생성된 각 파티션에 암호를 추가합니다. 이러한 암호를 /root 에 별도의 파일에 저장하고 --escrowcert 로 지정된 X.509 인증서를 사용하여 암호화됩니다. 이 옵션은 --escrowcert가 지정된 경우에만 의미가 있습니다.*

- *--PBKDF=PBKDF - LUKS 키 Iot에 대한 PBKDF(암호 기반 키 파생 기능) 알고리즘을 설정합니다. 도움말 페이지 cryptsetup(8) 도 참조하십시오. 이 옵션은 --encrypted가 지정된 경우에만 의미가 있습니다.*

- *--PBKDF-memory=PBKDF_MEMORY - PBKDF에 대한 메모리 비용을 설정합니다. 도움말 페이지 cryptsetup(8) 도 참조하십시오. 이 옵션은 --encrypted가 지정된 경우에만 의미가 있습니다.*

- *--PBKDF-time=PBKDF_TIME - PBKDF 암호 처리와 함께 사용할 밀리초 수를 설정합니다. man 페이지 cryptsetup(8) 에서도 --iter-time 을 참조하십시오. 이 옵션은 --encrypted 가 지정되고 --pbkdf-iterations 와 상호 배타적인 경우에만 의미가 있습니다.*

- *--PBKDF-iterations=PBKDF_ITERATIONS - 반복 횟수를 직접 설정하고 PBKDF 벤치마크를 방지합니다. man 페이지 cryptsetup(8) 에서 --pbkdf-force-iterations 도 참조하십시오. 이 옵션은 --encrypted 가 지정되고 --pbkdf-time 과 상호 배타적인 경우에만 의미가 있습니다.*

- *--resize= - 기존 파티션의 크기를 조정합니다. 이 옵션을 사용하는 경우 --size= 옵션과 --onpart= 옵션을 사용하여 대상 크기(MiB)를 지정합니다.*

참고

- *part 명령은 필수는 아니지만 Kickstart 스크립트에 부분,auto part 또는 mount 를 포함해야 합니다.*

- *Red Hat Enterprise Linux 8에서는 --active 옵션이 더 이상 사용되지 않습니다.*

- 어떤 이유로든 분할이 실패하면 진단 메시지가 가상 콘솔 **3**에 표시됩니다.

- **--noformat** 및 **--onpart** 를 사용하지 않는 한 생성된 모든 파티션은 설치 프로세스의 일부로 포맷됩니다.

- **sdX** (또는 **/dev/sdX**) 형식의 장치 이름은 재부팅 시 일관되게 유지되므로 일부 **Kickstart** 명령을 사용할 수 있습니다. 명령에서 장치 노드 이름을 호출할 때 **/dev/disk**의 모든 항목을 대신 사용할 수 있습니다. 예를 들면 다음과 같습니다.

```
part / --fstype=xfs --onpart=sda1
```

다음 중 하나와 유사한 항목을 사용할 수 있습니다.

```
part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
```

```
part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1
```

이렇게 하면 명령이 항상 동일한 스토리지 장치를 대상으로 합니다. 이는 특히 대규모 스토리지 환경에서 유용합니다. 스토리지 장치를 지속적으로 참조하기 위한 다양한 방법에 대한 자세한 내용은 스토리지 장치 관리 문서의 영구 이름 지정 장을 참조하십시오.

- **LUKS** 암호를 분실하는 경우 암호화된 파티션과 해당 데이터에 완전히 액세스할 수 없습니다. 분실된 암호를 복구할 방법은 없습니다. 그러나 **--escrowcert**를 사용하여 암호화 암호를 저장하고 **--backuppassphrase** 옵션을 사용하여 백업 암호화 암호를 생성할 수 있습니다.

B.5.14. RAID

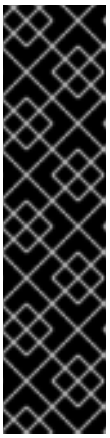
raid Kickstart 명령은 선택 사항입니다. 소프트웨어 **RAID** 장치를 어셈블합니다.

구문

```
raid mntpoint --level=level --device=device-name partitions*
```

옵션

- ***mntpoint - RAID** 파일 시스템이 마운트된 위치입니다. / 이면 부팅 파티션(/boot)이 없으면 **RAID** 수준이 **1**이어야 합니다. 부팅 파티션이 있는 경우 **/boot** 파티션은 수준 **1**이어야 하며 루트 (/) 파티션은 사용 가능한 유형 중 하나일 수 있습니다. **partitions*** (여러 파티션을 나열할 수 있음을 나타내는)에는 **RAID** 배열에 추가할 **RAID** 식별자가 나열됩니다.*



중요

- ***IBM Power Systems**에서 **RAID** 장치가 준비되어 설치 중에 다시 포맷되지 않은 경우, **/boot** 및 **PReP** 파티션을 **RAID** 장치에 배치하려는 경우 **RAID** 메타데이터 버전이 **0.90** 인지 확인합니다. 기본 **Red Hat Enterprise Linux 7 mdadm** 메타데이터 버전은 부팅 장치에 지원되지 않습니다.*

- ***PowerNV** 시스템에는 **PReP** 부트 파티션이 필요하지 않습니다.*

- ***--level= - RAID** 레벨은 **(0, 1, 4, 5, 6, 10)**입니다.*

  *사용 가능한 다양한 **RAID** 수준에 대한 정보는 **C.3절. “지원되는 RAID 유형”** 를 참조하십시오.*

- ***--device= -** 사용할 **RAID** 장치의 이름(예: **--device=root**).*

중요

***mdraid** 이름을 **md0** 형식으로 사용하지 마십시오. 이러한 이름은 영구적일 수 없습니다. 대신 루트 또는 스왑 과 같은 의미 있는 이름을 사용하십시오. 의미 있는 이름을 사용하면 **/dev/md/name** 에서 **/dev/mdX** 노드가 배열에 할당되는 심볼릭 링크가 생성됩니다.*

*이름을 지정할 수 없는 이전(**v0.90** 메타데이터) 배열이 있는 경우 파일 시스템 레이블 또는 **UUID**로 배열을 지정할 수 있습니다. 예를 들어 **--device=LABEL=root** 또는 **--device=UUID=93348e56-4631-d0f0-6f5b-45c47f570b88.***

RAID** 장치 자체의 **RAID** 장치 또는 **UUID**에서 파일 시스템의 **UUID**를 사용할 수 있습니다. **RAID** 장치의 **UUID**는 **8-4-4-4-12** 형식이어야 합니다. **mdadm**에서 보고한 **UUID**는 **8:8:8:8** 형식으로 되어 있어 변경해야 합니다. 예: **93348e56:4631d0f0:6f5b45c4:7f5b70b88** 은 **93348e56-4631-d0f0-6f5b47f547f570b88.

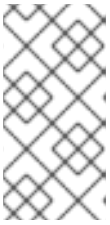
- ***--CHUNKSIZE= - RAID** 스토리지의 청크 크기를 **KiB**로 설정합니다. 특정 상황에서는 기본값과 다른 청크 크기(**512 Kib**)를 사용하면 **RAID**의 성능을 향상시킬 수 있습니다.*

- ***--spares= - RAID** 배열에 할당된 예비 드라이브 수를 지정합니다. 스페어 드라이브는 드라이브 장애가 발생할 경우 배열을 다시 작성하는 데 사용됩니다.*

- ***--fsprofile= -** 이 파티션에서 파일 시스템을 만드는 프로그램에 전달할 사용 유형을 지정합니다. 사용 유형은 파일 시스템을 만들 때 사용할 다양한 튜닝 매개 변수를 정의합니다. 이 옵션을 사용하려면 파일 시스템에서 사용 유형의 개념을 지원해야 하며 유효한 유형을 나열하는 구성 파일이 있어야 합니다. **ext2, ext3** 및 **ext4**의 경우 이 설정 파일은 **/etc/mke2fs.conf** 입니다.*

- ***--fstype= - RAID** 배열의 파일 시스템 유형을 설정합니다. 유효한 값은 **xfs,ext2,ext3,ext4,swap, vfat** 입니다.*

- ***--fsoptions= -** 파일 시스템을 마운트할 때 사용할 자유 형식의 옵션 문자열을 지정합니다. 이 문자열은 설치된 시스템의 **/etc/fstab** 파일에 복사되며 따옴표로 묶어야 합니다.*



참고

EFI 시스템 파티션(/boot/efi)에서 anaconda 하드 코딩은 값을 코딩하고 지정된 --fsoptions 값을 무시합니다.

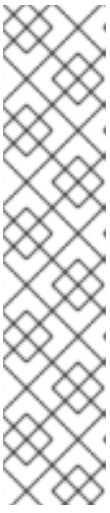
- *--mkfsoptions= - 이 파티션에서 파일 시스템을 만드는 프로그램에 전달할 추가 매개변수를 지정합니다. 인수 목록에서 처리가 수행되지 않으므로 mkfs 프로그램에 직접 전달할 수 있는 형식으로 제공해야 합니다. 즉, 파일 시스템에 따라 쉼표로 구분되거나 큰따옴표로 묶어야 합니다.*

- *--label= - 만들 파일 시스템에 제공할 라벨을 지정합니다. 지정된 레이블이 다른 파일 시스템에서 이미 사용 중인 경우 새 레이블이 생성됩니다.*

- *--noformat - 기존 RAID 장치를 사용하고 RAID 배열을 포맷하지 않습니다.*

- *--useexisting - 기존 RAID 장치를 사용하고 다시 포맷합니다.*

- *--encrypted - 이 RAID 장치를 --passphrase 옵션에 제공된 암호를 사용하여 LUKS(Linux Unified Key Setup)로 암호화하도록 지정합니다. 암호를 지정하지 않으면 Anaconda에서 autopart --passphrase 명령으로 설정된 기본 시스템 전체 암호를 사용하거나 설치를 중지하고 기본값이 설정되지 않은 경우 암호를 입력하라는 메시지가 표시됩니다.*



참고

하나 이상의 파티션을 암호화할 때 Anaconda는 256비트의 엔트로피를 수집하여 파티션을 안전하게 암호화하려고 합니다. 엔트로피 수집에는 약간의 시간이 걸릴 수 있습니다. 충분한 엔트로피가 수집되었는지에 관계없이 프로세스가 최대 10분 후에 중지됩니다.

이 프로세스는 설치 시스템과 상호 작용하여 정지할 수 있습니다(키보드에서 이동 또는 마우스 이동). 가상 머신에 설치하는 경우 virtio-rng 장치(가상 임의 번호 생성기)를 게스트에 연결할 수도 있습니다.

- *--LUKS-version=LUKS_VERSION - 파일 시스템을 암호화하는 데 사용할 LUKS 형식의 버전을 지정합니다. 이 옵션은 --encrypted가 지정된 경우에만 의미가 있습니다.*

- *--cipher= - Anaconda 기본 aes-xts-plain64 가 만족스럽지 않은 경우 사용할 암호화 유형*

*을 지정합니다. 이 옵션을 **--encrypted** 옵션과 함께 사용해야 합니다. 이 옵션 자체는 적용되지 않습니다. 사용 가능한 암호화 유형은 보안 강화 문서에 나열되어 있지만 **aes-xts-plain64** 또는 **aes-cbc-essiv:sha256** 을 사용하는 것이 좋습니다.*

- *--passphrase= - 이 **RAID** 장치를 암호화할 때 사용할 암호를 지정합니다. 이 옵션을 **--encrypted** 옵션과 함께 사용해야 합니다. 이 옵션 자체는 적용되지 않습니다.*

- *--escrowcert= URL_of_X.509_certificate - 이 장치에 대한 데이터 암호화 키를 **/root** 의 파일에 저장하고, **URL_of_X.509_certificate**로 지정된 **URL**에서 **X.509** 인증서를 사용하여 암호화됩니다. 이 옵션은 **--encrypted**가 지정된 경우에만 의미가 있습니다.*

- *--backuppassphrase - 무작위로 생성된 암호를 이 장치에 추가합니다. **/root** 의 파일에 암호를 저장하고 **--escrowcert** 로 지정된 **X.509** 인증서를 사용하여 암호화됩니다. 이 옵션은 **--escrowcert**가 지정된 경우에만 의미가 있습니다.*

- *--PBKDF=PBKDF - LUKS 키 lot에 대한 PBKDF(암호 기반 키 파생 기능) 알고리즘을 설정합니다. 도움말 페이지 **cryptsetup(8)** 도 참조하십시오. 이 옵션은 **--encrypted**가 지정된 경우에만 의미가 있습니다.*

- *--PBKDF-memory=PBKDF_MEMORY - PBKDF에 대한 메모리 비용을 설정합니다. 도움말 페이지 **cryptsetup(8)** 도 참조하십시오. 이 옵션은 **--encrypted**가 지정된 경우에만 의미가 있습니다.*

- *--PBKDF-time=PBKDF_TIME - **PBKDF** 암호 처리와 함께 사용할 밀리초 수를 설정합니다. **man** 페이지 **cryptsetup(8)** 에서도 **--iter-time** 을 참조하십시오. 이 옵션은 **--encrypted** 가 지정되고 **--pbkdf-iterations** 와 상호 배타적인 경우에만 의미가 있습니다.*

- *--PBKDF-iterations=PBKDF_ITERATIONS - 반복 횟수를 직접 설정하고 **PBKDF** 벤치마크를 방지합니다. **man** 페이지 **cryptsetup(8)** 에서 **--pbkdf-force-iterations** 도 참조하십시오. 이 옵션은 **--encrypted** 가 지정되고 **--pbkdf-time** 과 상호 배타적인 경우에만 의미가 있습니다.*

예제

*다음 예에서는 시스템에 **3**개의 **SCSI** 디스크가 있다고 가정하여 **/ home**에 대한 **RAID** 레벨 **1** 파티션과 **/home** 에 대한 **RAID** 수준 **5**를 생성하는 방법을 보여줍니다. 또한 각 드라이브에 하나씩 세 개의 스왑 파티션을 만듭니다.*

```
part raid.01 --size=6000 --ondisk=sda
part raid.02 --size=6000 --ondisk=sdb
part raid.03 --size=6000 --ondisk=sdc
```

```
part swap --size=512 --ondisk=sda
part swap --size=512 --ondisk=sdb
part swap --size=512 --ondisk=sdc
part raid.11 --size=1 --grow --ondisk=sda
part raid.12 --size=1 --grow --ondisk=sdb
part raid.13 --size=1 --grow --ondisk=sdc
raid / --level=1 --device=rhel8-root --label=rhel8-root raid.01 raid.02 raid.03
raid /home --level=5 --device=rhel8-home --label=rhel8-home raid.11 raid.12 raid.13
```

참고

- **LUKS** 암호를 분실하는 경우 암호화된 파티션과 해당 데이터에 완전히 액세스할 수 없습니다. 분실된 암호를 복구할 방법은 없습니다. 그러나 **--escrowcert**를 사용하여 암호화 암호를 저장하고 **--backuppassphrase** 옵션을 사용하여 백업 암호화 암호를 생성할 수 있습니다.

B.5.15. reqpart

reqpart Kickstart 명령은 선택 사항입니다. 하드웨어 플랫폼에 필요한 파티션을 자동으로 생성합니다. 여기에는 **UEFI** 펌웨어가 있는 시스템의 **/boot/efi** 파티션, **BIOS** 펌웨어 및 **GPT**가 있는 시스템의 **BIOS** 부팅 파티션, **IBM Power Systems**용 **PRePBoot** 파티션이 포함됩니다.

구문

```
reqpart [--add-boot]
```

옵션

- **--add-boot - base** 명령으로 생성된 플랫폼별 파티션 외에도 별도의 **/boot** 파티션을 만듭니다.

참고

- **autopart** 는 **reqpart** 명령이 수행하는 모든 작업을 수행하고 또한 **/** 및 **swap** 과 같은 다른 파티션 또는 논리 볼륨을 생성하므로 이 명령은 **autopart** 와 함께 사용할 수 없습니다. **autopart** 와 달리 이 명령은 플랫폼별 파티션만 생성하고 나머지 드라이브는 빈 상태로 유지되므로 사용자 지정 레이아웃을 만들 수 있습니다.

B.5.16. 스냅샷

*스냅샷 **Kickstart** 명령은 선택 사항입니다. 이 스냅샷을 사용하여 설치 프로세스 중에 **LVM** 씬 볼륨 스냅샷을 생성합니다. 이렇게 하면 설치 전이나 후에 논리 볼륨을 백업할 수 있습니다.*

*여러 스냅샷을 생성하려면 **snaphost Kickstart** 명령을 여러 번 추가합니다.*

구문

```
snapshot vg_name/lv_name --name=snapshot_name --when=pre-install|post-install
```

옵션

- ***VG_NAME /lv_name -** 볼륨 그룹 및 논리 볼륨의 이름을 설정하여 스냅샷을 만듭니다.*
- ***--name=snapshot_name -** 스냅샷 이름을 설정합니다. 이 이름은 볼륨 그룹 내에서 고유해야 합니다.*
- ***--when=pre-install|post-install-** 설치가 시작되기 전에 스냅샷을 만드는 경우 또는 설치가 완료된 후 설정합니다.*

B.5.17. volgroup

***volgroup Kickstart** 명령은 선택 사항입니다. **LVM(Logical Volume Management)** 그룹을 생성합니다.*

구문

```
volgroup name [OPTIONS] [partition*]
```

필수 옵션

- ***name -** 새 볼륨 그룹의 이름입니다.*

옵션

- *파티션 **-** 볼륨 그룹의 백업 스토리지로 사용할 물리 볼륨 파티션입니다.*

- ***--noformat -** 기존 볼륨 그룹을 사용하고 포맷하지 마십시오.*

- ***--useexisting -** 기존 볼륨 그룹을 사용하고 다시 포맷합니다. 이 옵션을 사용하는 경우 파티션을 지정하지 마십시오. 예를 들어 다음과 같습니다.*

```
volgroup rhel00 --useexisting --noformat
```

- ***--pesize= -** 볼륨 그룹의 물리 확장 영역 크기를 **KiB**로 설정합니다. 기본값은 **4096(4MiB)**이고, 최소 값은 **1024(1MiB)**입니다.*

- ***--reserved-space= -** 볼륨 그룹에서 사용하지 않는 공간**(MiB)**을 지정합니다. 새로 생성된 볼륨 그룹에만 적용됩니다.*

- ***--reserved-percent= -** 사용하지 않은 상태로 남겨 둘 총 볼륨 그룹 공간을 지정합니다. 새로 생성된 볼륨 그룹에만 적용됩니다.*

참고

- *먼저 파티션을 만든 다음 논리 볼륨 그룹을 만든 다음 논리 볼륨을 만듭니다. 예를 들어 다음과 같습니다.*

```
part pv.01 --size 10000
volgroup my_volgrp pv.01
logvol / --vgname=my_volgrp --size=2000 --name=root
```

- ***Kickstart**를 사용하여 **Red Hat Enterprise Linux**를 설치할 때 논리 볼륨 및 볼륨 그룹 이름에 대시**(-)** 문자를 사용하지 마십시오. 이 문자를 사용하는 경우 설치가 정상적으로 완료되지만 **/dev/mapper/** 디렉터리에 모든 대시가 두 배가 되어 이러한 볼륨과 볼륨 그룹이 나열됩니다. 예를 들어 **logvol-01** 이라는 논리 볼륨이 포함된 **volgrp-01** 이라는 볼륨 그룹이 **/dev/mapper/volp--01-logvol--01** 으로 나열됩니다.*

*이 제한은 새로 생성된 논리 볼륨 및 볼륨 그룹 이름에만 적용됩니다. **--noformat** 옵션을 사용하여 기존 항목을 재사용하는 경우 해당 이름은 변경되지 않습니다.*

B.5.18. zerombr

***zerombr Kickstart** 명령은 선택 사항입니다. **zerombr** 는 디스크에 있는 잘못된 파티션 테이블을 초기화하고 잘못된 파티션 테이블이 있는 디스크의 모든 내용을 삭제합니다. 이 명령은 포맷되지 않은 **Direct Access Storage Device(DASD)** 디스크를 사용하여 **64**비트 **IBM Z** 시스템에서 설치를 수행해야 합니다. 그렇지 않으면 포맷되지 않은 디스크가 설치 중에 포맷되어 사용되지 않습니다.*

구문

```
zerombr
```

참고

- *64비트 **IBM Z**에서 **0mbr** 이 지정되면, 이미 낮은 수준의 형식이 아닌 설치 프로그램에 표시되는 모든 **Direct Access Storage Device(DASD)**는 **dasdfmt**로 자동으로 하위 수준 형식으로 지정됩니다. 또한 이 명령은 대화식 설치 중에 사용자 선택을 방지합니다.*
- *__zerombr__ 을 지정하지 않고 설치 프로그램에 표시되는 포맷되지 않은 **DASD**가 하나 이상 있으면 비대화형 **Kickstart** 설치가 실패합니다.*
- *__zerombr__ 을 지정하지 않고 설치 프로그램에 표시되는 포맷되지 않은 **DASD**가 하나 이상 있는 경우 사용자가 표시된 모든 **DASD**를 포맷하는 데 동의하지 않으면 대화형 설치가 종료됩니다. 이를 우회하려면 설치 중에 사용할 **DASD**만 활성화합니다. 설치가 완료된 후 항상 더 많은 **DASD**를 추가할 수 있습니다.*
- *이 명령에는 옵션이 없습니다.*

B.5.19. zfcp

***zfcp Kickstart** 명령은 선택 사항입니다. 파이버 채널 장치를 정의합니다.*

이 옵션은 64비트 IBM Z에만 적용됩니다. 아래에 설명된 모든 옵션을 지정해야 합니다.

구문

```
zfcp --devnum=devnum --wwpn=wwpn --fcplun=lun
```

옵션

- --devnum= - 장치 번호(zFCP 어댑터 장치 버스 ID)입니다.
- --WWPN= - 장치의 WWWN(World Wide Port Name)입니다. 16자리 숫자의 형식을 사용합니다. 앞에 0x 가 있습니다.
- --fcplun= - 장치의 논리 단위 번호(LUN)입니다. 16자리 숫자의 형식을 사용합니다. 앞에 0x 가 있습니다.

예제

```
zfcp --devnum=0.0.4000 --wwpn=0x5005076300C213e9 --fcplun=0x5022000000000000
```

B.6. RHEL 설치 프로그램과 함께 제공되는 애드온을 위한 KICKSTART 명령

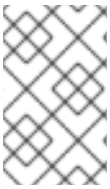
이 섹션의 Kickstart 명령은 기본적으로 Red Hat Enterprise Linux 설치 프로그램인 Kdump 및 OpenSCAP와 함께 제공되는 애드온과 관련이 있습니다.

B.6.1. %addon com_redhat_kdump

%addon com_redhat_kdump Kickstart 명령은 선택 사항입니다. 이 명령은 kdump 커널 크래시 덤프 메커니즘을 구성합니다.

구문

```
%addon com_redhat_kdump [OPTIONS]
%end
```



참고

*이 명령의 구문은 기본 제공 **Kickstart** 명령이 아닌 추가 기능이므로 드물지 않습니다.*

참고

***kdump**는 나중에 분석을 위해 시스템의 메모리 내용을 저장할 수 있는 커널 크래시 덤프 메커니즘입니다. 시스템을 재부팅하지 않고 다른 커널의 컨텍스트에서 **Linux** 커널을 부팅하는 데 사용할 수 있는 **kexec** 를 사용하고, 손실되는 첫 번째 커널 메모리의 내용을 보존합니다.*

*시스템 충돌의 경우 **kexec** 는 두 번째 커널(**capture kernel**)으로 부팅됩니다. 이 캡처 커널은 시스템 메모리의 예약된 부분에 있습니다. 그런 다음 **kdump**는 충돌한 커널의 메모리 (**crash dump**)의 내용을 캡처하여 지정된 위치에 저장합니다. 이 **Kickstart** 명령을 사용하여 위치를 구성할 수 없습니다. **/etc/kdump.conf** 구성 파일을 편집하여 설치 후 구성해야 합니다.*

***Kdump**에 대한 자세한 내용은 커널 문서 관리, 모니터링 및 업데이트의 **kdump** 설치 장을 참조하십시오.*

옵션

- ***--enable -** 설치된 시스템에서 **kdump** 활성화.*
- ***--disable -** 설치된 시스템에서 **kdump**를 비활성화합니다.*
- ***--Reserve-mb= - kdump**를 위해 예약할 메모리 양(**MiB**)입니다. 예를 들어 다음과 같습니다.*

```
%addon com_redhat_kdump --enable --reserve-mb=128
%end
```

숫자 값 대신 **auto** 를 지정할 수도 있습니다. 이 경우 설치 프로그램은 커널 문서 관리, 모니터링 및 업데이트의 **kdump** 섹션에 설명된 기준에 따라 자동으로 메모리 양을 결정합니다.

kdump를 활성화하고 **--reserve-mb=** 옵션을 지정하지 않으면 **auto** 값이 사용됩니다.

- **--enablefadump** - 이를 허용하는 시스템에서 펌웨어 지원 덤프를 활성화합니다(특히 **IBM Power Systems** 서버).

B.6.2. %addon org_fedora_oscap

%addon org_fedora_oscap Kickstart 명령은 선택 사항입니다.

OpenSCAP 설치 프로그램 애드온은 설치된 시스템에 **SCAP**(보안 콘텐츠 자동화 프로토콜) 콘텐츠를 적용하는 데 사용됩니다. 이 애드온은 **Red Hat Enterprise Linux 7.2** 이후 기본적으로 활성화되어 있습니다. 활성화하면 이 기능을 제공하는 데 필요한 패키지가 자동으로 설치됩니다. 그러나 기본적으로 정책은 적용되지 않습니다. 즉, 특별히 구성하지 않는 한 설치 중 또는 설치 후 검사가 수행되지 않습니다.

중요

모든 시스템에서 보안 정책을 적용할 필요는 없습니다. 이 명령은 조직 규칙 또는 정부 규정에서 특정 정책을 준수하는 경우에만 사용해야 합니다.

대부분의 다른 명령과 달리 이 애드온은 일반 옵션을 허용하지 않지만 **%addon** 정의 본문에 키-값 쌍을 대신 사용합니다. 이러한 쌍은 공백과 무관합니다. 값은 선택적으로 작은따옴표(') 또는 큰따옴표(")로 묶을 수 있습니다.

구문

```
%addon org_fedora_oscap
key = value
%end
```

키

다음 키는 애드온에 의해 인식됩니다.

content-type

보안 콘텐츠의 유형입니다. 가능한 값은 **datastream,아카이브,rpm, scap-security-guide** 입니다.

content-type 이 **scap-security-guide** 이면 애드온은 부팅 미디어에 있는 **scap-security-guide** 패키지에서 제공하는 콘텐츠를 사용합니다. 즉, 프로필을 제외한 다른 모든 키는 아무런 영향을 미치지 않습니다.

content-url

보안 콘텐츠의 위치입니다. **HTTP, HTTPS** 또는 **FTP**를 사용하여 콘텐츠에 액세스할 수 있어야 합니다. 현재 로컬 스토리지는 지원되지 않습니다. 원격 위치의 콘텐츠 정의에 연결하려면 네트워크 연결을 사용할 수 있어야 합니다.

datastream-id

content-url 값에 참조되는 데이터 스트림의 **ID**입니다. **content-type** 이 데이터 스트림 인 경우에만 사용됩니다.

xccdf-id

사용하려는 벤치마크의 **ID**입니다.

content-path

데이터 스트림 경로 또는 사용할 **XCCDF** 파일의 경로(아카이브에서 상대 경로로 지정됨).

profile

적용할 프로필의 **ID**입니다. **default** 를 사용하여 **default** 프로필을 적용합니다.

fingerprint

content-url 에서 참조하는 콘텐츠의 **MD5, SHA1** 또는 **SHA2** 체크섬입니다.

tailoring-path

아카이브에서 상대 경로로 지정되는 맞춤형 파일의 경로입니다.

예제

- 다음은 설치 미디어의 **scap-security-guide** 의 콘텐츠를 사용하는 **%addon**

org_fedora_oscap 섹션 예입니다.

예 B.1. SCAP 보안 가이드를 사용하여 OpenSCAP 애드온 정의 샘플

```
%addon org_fedora_oscap
content-type = scap-security-guide
profile = xccdf_org.ssgproject.content_profile_pci-dss
%end
```

- *다음은 웹 서버에서 사용자 지정 프로필을 로드하는 더 복잡한 예입니다.*

예 B.2. 데이터 스트림을 사용한 샘플 OpenSCAP 애드온 정의

```
%addon org_fedora_oscap
content-type = datastream
content-url = http://www.example.com/scap/testing_ds.xml
datastream-id = scap_example.com_datastream_testing
xccdf-id = scap_example.com_cref_xccdf.xml
profile =  xccdf_example.com_profile_my_profile
fingerprint = 240f2f18222faa98856c3b4fc50c4195
%end
```

추가 리소스

- *Security Hardening*
- *OpenSCAP 설치 프로그램 애드온*
- *OpenSCAP Portal*

B.7. ANACONDA에 사용되는 명령

pwpolicy 명령은 Kickstart 파일의 %anaconda 섹션에서만 사용할 수 있는 Anaconda UI 특정 명령입니다.

B.7.1. pwpolicy

pwpolicy Kickstart 명령은 선택 사항입니다. 설치 중에 사용자 지정 암호 정책을 적용하려면 이 명령을 사용합니다. 이 정책을 사용하려면 root, 사용자 또는 luks 사용자 계정에 대한 암호를 생성해야 합니

다. 암호 길이 및 힘과 같은 요인은 암호의 유효성을 결정합니다.

구문

```
pwpolicy name [--minlen=length] [--minquality=quality] [--strict|--nostrict] [--emptyok|--noempty] [--changesok|--nochanges]
```

필수 옵션

- **name -** 루트,사용자 또는 **luks** 로 교체하여 루트 암호, 사용자 암호 또는 **LUKS** 암호에 대한 정책을 적용합니다.

선택적 옵션

- **--minlen= -** 허용된 최소 암호 길이를 문자 단위로 설정합니다. 기본값은 **6** 입니다.
- **--minquality= - lib Replicas** 라이브러리에서 정의한 대로 허용된 최소 암호 품질을 설정합니다. 기본값은 **1** 입니다.
- **--strict -** 엄격한 암호 적용 활성화. **--minquality=** 및 **--minlen=** 에 지정된 요구 사항을 충족하지 않는 암호는 허용되지 않습니다. 이 옵션은 기본적으로 비활성화되어 있습니다.
- **--notstrict - - minquality=** 및 **-minlen=** 옵션에 의해 지정된 최소 품질 요구 사항을 충족하지 않는 암호가 허용됩니다. 텍스트 모드 인터페이스의 경우 비슷한 메커니즘이 사용됩니다.
- **--emptyok -** 빈 암호를 사용할 수 있습니다. 사용자 암호에 대해 기본적으로 활성화되어 있습니다.
- **--notempty -** 비어 있는 암호 사용을 비활성화합니다. 루트 암호 및 **LUKS** 암호에 대해 기본적으로 활성화되어 있습니다.
- **--changesok - Kickstart** 파일에서 이미 암호를 지정하는 경우에도 사용자 인터페이스에서 암호를 변경할 수 있습니다. 기본적으로 비활성되어 있습니다.

- *--nochanges - Kickstart 파일에 이미 설정된 암호 변경을 허용합니다. 기본적으로 활성화되어 있습니다.*

참고

- *pwpolicy 명령은 Kickstart 파일의 %anaconda 섹션에서만 사용할 수 있는 Anaconda-UI 특정 명령입니다.*

- *libResourceOverride 라이브러리는 최소 암호 요구 사항(length 및 quality)을 확인하는 데 사용됩니다. lib ResourceOverride 패키지에서 제공하는 pwscore 및 pwmake 명령을 사용하여 암호의 품질 점수를 확인하거나 지정된 점수로 임의의 암호를 생성할 수 있습니다. 이러한 명령에 대한 자세한 내용은 pwscore(1) 및 pwmake(1) 매뉴얼 페이지를 참조하십시오.*

B.8. 시스템 복구를 위한 KICKSTART 명령

이 섹션의 Kickstart 명령은 설치된 시스템을 복구합니다.

B.8.1. 구조

복구 Kickstart 명령은 선택 사항입니다. 쉘 환경에 root 권한 및 일련의 시스템 관리 툴을 제공하여 설치를 복구하고 다음과 같은 문제를 해결합니다.

- *파일 시스템을 읽기 전용으로 마운트*
- *드라이버 디스크에 제공된 드라이버 추가 또는 차단 목록 추가*
- *시스템 패키지 설치 또는 업그레이드*
- *파티션 관리*

참고

Kickstart 복구 모드는 **systemd** 및 서비스 관리자의 일부로 제공되는 복구 모드 및 긴급 모드와 다릅니다.

rescue 명령은 시스템을 자체적으로 수정하지 않습니다. 읽기-쓰기 모드에서 **/mnt/sysimage** 아래에 시스템을 마운트하여 복구 환경만 설정합니다. 시스템을 마운트하지 않거나 읽기 전용 모드로 마운트할 수 있습니다.

구문

```
rescue [--nomount|--romount]
```

옵션

- **--nomount** 또는 **--romount** - 설치된 시스템이 복구 환경에서 마운트되는 방법을 제어합니다. 기본적으로 설치 프로그램은 시스템을 찾아서 읽기-쓰기 모드로 마운트하고 이 마운트를 수행한 위치를 알려줍니다. 선택적으로 아무것도 마운트하지 않도록 선택하거나(**- nomount** 옵션) 읽기 전용 모드(**- romount** 옵션)로 마운트할 수 있습니다. 이 두 가지 옵션 중 하나만 사용할 수 있습니다.

참고

복구 모드를 실행하려면 **Kickstart** 파일의 사본을 만들고 여기에 **rescue** 명령을 포함합니다.

rescue 명령을 사용하면 설치 프로그램이 다음 단계를 수행합니다.

1. **%pre** 스크립트를 실행합니다.
2. 복구 모드를 위한 환경을 설정합니다.

   다음 **Kickstart** 명령이 적용됩니다.

a. *업데이트*

b. ***sshpw***

c. ***logging***

d. ***lang***

e. ***network***

3. *고급 스토리지 환경을 설정합니다.*

   *다음 **Kickstart** 명령이 적용됩니다.*

a. ***fcoe***

b. ***iscsi***

c. ***iscsiname***

d. ***nvdimm***

e. ***zfcp***

4. *시스템 마운트*

```
rescue [--nomount|--romount]
```

5. ***%post** 스크립트 실행*

이 단계는 설치된 시스템이 읽기-쓰기 모드로 마운트된 경우에만 실행됩니다.

6. *쉘 시작*

7. *시스템 재부팅*

부록 C. 파티션 참조

C.1. 지원되는 장치 유형

표준 파티션

표준 파티션은 파일 시스템 또는 스왑 공간을 포함할 수 있습니다. 표준 파티션은 /boot 및 BIOS Boot 및 EFI 시스템 파티션에 가장 일반적으로 사용됩니다. 대부분의 다른 사용에 LVM 논리 볼륨을 사용하는 것이 좋습니다.

LVM

장치 유형으로 LVM (또는 논리 볼륨 관리)을 선택하면 LVM 논리 볼륨이 생성됩니다. LVM은 물리 디스크를 사용할 때 성능을 향상시키고 하나의 마운트 지점에 여러 물리 디스크를 사용하는 등 고급 설정을 허용하고 성능, 안정성 또는 둘 다 향상을 위해 소프트웨어 RAID를 설정할 수 있습니다.

LVM thin provisioning

씬 프로비저닝을 사용하면 씬 풀이라는 여유 공간 스토리지 풀을 관리할 수 있으며, 애플리케이션에 필요할 때 임의의 수의 장치에 할당할 수 있습니다. 스토리지 공간을 비용 효율적으로 할당하기 위해 필요한 경우 풀을 동적으로 확장할 수 있습니다.

주의

설치 프로그램은 프로비저닝된 LVM 씬 풀을 지원하지 않습니다.

C.2. 지원되는 파일 시스템

이 섹션에서는 Red Hat Enterprise Linux에서 사용 가능한 파일 시스템에 대해 설명합니다.

xfs

XFS 는 확장 가능한 고성능 파일 시스템으로, 최대 16개의 exabytes(약 16만TB) 파일, 최대 8개의 엑사바이트(약 8백만 테라바이트) 및 수만 개의 항목이 포함된 디렉토리 구조입니다. XFS 는 또한 더 빠른 충돌 복구를 용이하게 하는 메타데이터 저널링을 지원합니다. 단일 XFS 파일 시스템의 지원되는 최대 크기는 500TB입니다. XFS 는 Red Hat Enterprise Linux의 기본 및 권장 파일 시스템입니다. XFS 파일 시스템은 여유 공간을 얻기 위해 축소할 수 없습니다.

ext4

ext4 파일 시스템은 ext3 파일 시스템을 기반으로 하며 여러 개선사항을 제공합니다. 여기에는 더

큰 파일 시스템과 더 큰 파일 지원, 디스크 공간 할당, 디렉토리 내 하위 디렉터리 수에 대한 제한 없음, 더 빠른 파일 시스템 검사, 더 강력한 저널링이 포함됩니다. 단일 ext4 파일 시스템의 지원되는 최대 크기는 50TB입니다.

ext3

ext3 파일 시스템은 ext2 파일 시스템을 기반으로 하며 하나의 주요 이점인 저널링을 사용합니다. journaling 파일 시스템을 사용하면 항상 fsck 유틸리티를 실행하여 파일 시스템의 메타데이터 일관성을 확인할 필요가 없기 때문에 파일 시스템을 복구할 수 있는 시간을 줄일 수 있습니다.

ext2

ext2 파일 시스템은 일반 파일, 디렉터리 또는 심볼릭 링크를 포함한 표준 Unix 파일 유형을 지원합니다. 최대 255자까지 긴 파일 이름을 할당할 수 있습니다.

swap

스왑 파티션은 가상 메모리를 지원하는 데 사용됩니다. 즉, 시스템이 처리 중인 데이터를 저장할 RAM이 충분하지 않은 경우 데이터가 스왑 파티션에 기록됩니다.

vfat

VFAT 파일 시스템은 FAT 파일 시스템에서 Microsoft Windows 긴 파일 이름과 호환되는 Linux 파일 시스템입니다.

참고

VFAT 파일 시스템에 대한 지원은 Linux 시스템 파티션에 대해 사용할 수 없습니다. 예를 들면 /, /var,/usr 등입니다.

BIOS Boot

BIOS 시스템에서 GUID 파티션 테이블(GPT)이 있는 장치에서 부팅하는 데 매우 작은 파티션이 필요하고 BIOS 호환성 모드의 UEFI 시스템은 필요합니다.

EFI 시스템 파티션

UEFI 시스템에서 GUID 파티션 테이블(GPT)을 사용하여 장치를 부팅하는 데 필요한 작은 파티션입니다.

PReP

이 작은 부팅 파티션은 하드 드라이브의 첫 번째 파티션에 있습니다. PReP 부팅 파티션에는 다른 IBM Power Systems 서버가 Red Hat Enterprise Linux를 부팅할 수 있는 GRUB2 부트 로더가 포함되어 있습니다.

참고

- PowerNV 시스템에는 PReP 부트 파티션이 필요하지 않습니다.

C.3. 지원되는 RAID 유형

RAID는 여러 물리적 디스크를 논리 단위로 결합할 수 있는 기술입니다. 일부 설정은 신뢰성 비용으로 성능을 개선하도록 설계되었지만 다른 설정은 동일한 양의 사용 가능한 공간에 대해 더 많은 디스크를 요구하는 비용으로 안정성을 향상시킵니다.

이 섹션에서는 LVM 및 LVM Thin Provisioning과 함께 사용하여 설치된 시스템에 스토리지를 설정할 수 있는 지원되는 소프트웨어 RAID 유형에 대해 설명합니다.

RAID 0

성능: 여러 디스크에 데이터를 분산합니다. RAID 0은 표준 파티션보다 향상된 성능을 제공하며 여러 디스크의 스토리지를 하나의 대규모 가상 장치로 풀링하는 데 사용할 수 있습니다. RAID 0은 중복을 제공하지 않으며 배열에서 하나의 장치 오류가 전체 배열의 데이터를 제거한다는 점에 유의하십시오. RAID 0에는 디스크가 두 개 이상 필요합니다.

RAID 1

redundancy: 한 파티션에서 하나 이상의 다른 디스크에 있는 모든 데이터를 미러링합니다. 배열의 추가 장치는 중복성의 증가 수준을 제공합니다. RAID 1에는 디스크가 두 개 이상 필요합니다.

RAID 4

오류 확인: 여러 디스크에 데이터를 분리하고 배열의 한 디스크를 사용하여 배열의 디스크가 실패하는 경우 배열을 보호하는 패리티 정보를 저장합니다. 모든 패리티 정보가 하나의 디스크에 저장되므로 이 디스크에 대한 액세스가 배열의 성능에서 "bottleneck"을 생성합니다. RAID 4에는 최소 3개의 디스크가 필요합니다.

RAID 5

분산 오류 확인: 여러 디스크에 데이터와 패리티 정보를 분리합니다. RAID 5는 데이터를 여러 디스크에 분산시키는 성능 이점을 제공하지만 패리티 정보가 어레이를 통해 배포되므로 RAID 4의 성능 병목 현상은 공유하지 않습니다. RAID 5에는 최소 3개의 디스크가 필요합니다.

RAID 6

중복 오류 확인: RAID 6은 RAID 5와 유사하지만 하나의 패리티 데이터 세트만 저장하는 대신 두 세트를 저장합니다. RAID 6에는 최소 4개의 디스크가 필요합니다.

RAID 10

*성능 및 중복: **RAID 10**은 중첩 또는 하이브리드 **RAID**입니다. 미러링된 디스크 세트를 통해 데이터를 배포하여 구성됩니다. 예를 들어 **RAID 10** 배열은 **4**개의 **RAID** 파티션에서 구성된 두 개의 미러링된 파티션 쌍으로 구성됩니다. **RAID 10**에는 디스크가 **4**개 이상 필요합니다.*

C.4. 권장 파티션 스키마

*다음 마운트 지점에 별도의 파일 시스템을 생성하는 것이 좋습니다. 그러나 필요한 경우 **/usr,/var, /tmp** 마운트 지점에 파일 시스템을 생성할 수도 있습니다.*

- ***/boot***
- ***/ (root)***
- ***/home***
- ***swap***
- ***/boot/efi***
- ***PReP***

이 파티션 스키마는 베어 메탈 배포에 권장되며 가상 및 클라우드 배포에는 적용되지 않습니다.

/boot 파티션 - 권장 크기 1GiB

***/boot** 에 마운트된 파티션에는 운영 체제 커널이 포함되어 있어 시스템이 부트스트랩 프로세스 중에 사용되는 파일과 함께 **Red Hat Enterprise Linux 8**을 부팅할 수 있습니다. 대부분의 펌웨어의 제한으로 인해 이를 보관할 작은 파티션을 만드는 것이 좋습니다. 대부분의 시나리오에서는 **1GiB**의 부팅 파티션이 충분합니다. 다른 마운트 지점과 달리 **/boot** 에 **LVM** 볼륨을 사용할 수 없습니다. **/boot** 는 별도의 디스크 파티션에 있어야 합니다.*

주의

일반적으로 /boot 파티션은 설치 프로그램에 의해 자동으로 생성됩니다. 그러나 / (root) 파티션이 2TiB보다 크고 (U)EFI가 부팅에 사용되는 경우 시스템을 성공적으로 부팅하기 위해 2TiB보다 작은 별도의 /boot 파티션을 생성해야 합니다.

참고

RAID 카드가 있는 경우 일부 BIOS 유형이 RAID 카드에서 부팅을 지원하지 않는다는 점에 유의하십시오. 이러한 경우 /boot 파티션은 별도의 하드 드라이브와 같이 RAID 배열 외부의 파티션에 생성해야 합니다.

루트 - 권장 크기 10GiB

여기서 "/" 또는 루트 디렉터리가 있습니다. 루트 디렉터리는 디렉터리 구조의 최상위 수준입니다. 기본적으로 모든 파일은 다른 파일 시스템이 파일에 기록되는 경로(예: /boot 또는 /home)에 마운트되지 않는 한 이 파일 시스템에 기록됩니다.

5GiB 루트 파일 시스템에서 최소 설치를 설치할 수 있지만 원하는 만큼 많은 패키지 그룹을 설치할 수 있도록 최소 10GiB를 할당하는 것이 좋습니다.

중요

/ 디렉토리를 / root 디렉토리와 혼동하지 마십시오. /root 디렉토리는 root 사용자의 홈 디렉토리입니다. /root 디렉토리를 루트 디렉토리와 구분하기 위해 슬래시 루트라고도 합니다.

/home - 권장 크기 1GiB

시스템 데이터와 별도로 사용자 데이터를 저장하려면 /home 디렉토리에 대한 전용 파일 시스템을 생성합니다. 로컬로 저장되는 데이터 양, 사용자 수 등을 기준으로 파일 시스템 크기를 기반으로 합니다. 사용자 데이터 파일을 삭제하지 않고 Red Hat Enterprise Linux 8을 업그레이드하거나 다시 설치할 수 있습니다. 자동 파티션을 선택하면 설치에 사용 가능한 디스크 공간을 55GiB 이상 사용하여 /home 파일 시스템이 생성되었는지 확인하는 것이 좋습니다.

스왑 파티션 - 권장 크기 1GiB

스왑 파일 시스템은 가상 메모리를 지원합니다. 시스템이 처리 중인 데이터를 저장할 RAM이 충분하지 않은 경우 데이터는 스왑 파일 시스템에 기록됩니다. 스왑 크기는 총 시스템 메모리가 아닌 시

스템 메모리 워크로드의 함수이므로 총 시스템 메모리 크기와 동일하지 않습니다. 시스템이 실행 중인 애플리케이션을 분석하고 시스템 메모리 워크로드를 결정하기 위해 해당 애플리케이션을 로드하는 것이 중요합니다. 애플리케이션 공급자 및 개발자는 지침을 제공할 수 있습니다.

*시스템이 스왑 공간이 부족하면 시스템 **RAM** 메모리가 소진되어 커널이 프로세스를 종료합니다. 스왑 공간을 너무 많이 구성하면 스토리지 장치가 할당되지만 유휴 상태이며 리소스를 잘못 사용할 수 없습니다. 너무 많은 스왑 공간도 메모리 누수를 숨길 수 있습니다. 스왑 파티션의 최대 크기 및 기타 추가 정보는 **mkswap(8)** 매뉴얼 페이지에서 찾을 수 있습니다.*

*다음 표에서는 시스템의 **RAM** 크기에 따라 권장되는 스왑 파티션 크기를 제공하고 시스템에 충분한 메모리를 원하는 경우 최대 절전 상태로 설정할 수 있습니다. 설치 프로그램이 시스템을 자동으로 파티션하도록 설정하면 이러한 지침을 사용하여 스왑 파티션 크기가 설정됩니다. 자동 파티션 설정에서는 힙을 사용하지 않는 것으로 가정합니다. 스왑 파티션의 최대 크기는 하드 드라이브의 총 크기 중 **10 %**로 제한되며 설치 프로그램은 **1TiB** 이상의 스왑 파티션을 생성할 수 없습니다. 최대 절전 모드를 허용할 수 있는 스왑 공간을 설정하거나 스왑 파티션 크기를 시스템 스토리지 공간의 **10%** 이상으로 설정하려면 파티션 레이아웃을 수동으로 편집해야 합니다.*

*표 **C.1.** 권장되는 시스템 스왑 공간*

시스템의 RAM 크기	권장되는 스왑 공간	최대 절전을 허용하는 경우 권장 스왑 공간
2GiB 미만	RAM의 두 배 크기	3배 RAM 용량
2 GiB - 8 GiB	RAM의 양과 같습니다.	RAM의 두 배 크기
8GiB - 64GiB	4GiB ~ 0.5배 RAM 용량	1.5배 RAM 용량
64GiB 이상	워크로드 종속(최소 4GiB)	Hibernation이 권장되지 않음

/boot/efi** 파티션 **-** 권장 크기 **200MiB

***UEFI** 기반 **AMD64, Intel 64** 및 **64**비트 **ARM**에는 **200MiB EFI** 시스템 파티션이 필요합니다. 최소 최소 크기는 **200MiB**이고 기본 크기는 **600MiB**이고, 최대 크기는 **600MiB**입니다. **BIOS** 시스템에는 **EFI** 시스템 파티션이 필요하지 않습니다.*

*각 범위 간의 경계(예: **2GiB, 8GiB** 또는 **64GiB** 시스템 **RAM**이 있는 시스템)에서 선택한 스왑 공간과 하이퍼션 지원과 관련하여 재량을 행사할 수 있습니다. 시스템 리소스에서 허용하는 경우 스왑 공간을 늘리면 성능이 향상될 수 있습니다.*

특히 빠른 드라이브, 컨트롤러 및 인터페이스가 있는 시스템에서 여러 스토리지 장치에 스왑 공간을 분산하면 스왑 공간 성능이 향상됩니다.

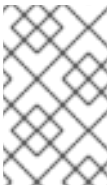
대부분의 시스템에는 필요한 최소 것보다 많은 파티션과 볼륨이 있습니다. 특정 시스템 요구에 따라 파티션을 선택합니다.

참고

- 즉시 필요한 파티션에 스토리지 용량을 할당합니다. 언제든지 여유 공간을 할당하여 발생하는 대로 요구 사항을 충족할 수 있습니다.
- 파티션을 구성하는 방법을 잘 모르는 경우 설치 프로그램에서 제공하는 자동 기본 파티션 레이아웃을 수락합니다.

부팅 파티션 준비 - 4에서 8MiB의 권장 크기

IBM Power System 서버에 **Red Hat Enterprise Linux**를 설치할 때 하드 드라이브의 첫 번째 파티션에 **PReP** 부팅 파티션이 포함되어야 합니다. 여기에는 다른 **IBM Power Systems** 서버가 **Red Hat Enterprise Linux**를 부팅할 수 있는 **GRUB2** 부트 로더가 포함되어 있습니다.



참고

- **PowerNV** 시스템에는 **PReP** 부트 파티션이 필요하지 않습니다.

C.5. 파티션 관련 설명

모든 시스템을 분할하는 가장 좋은 방법은 없습니다. 최적의 설정은 설치 중인 시스템을 사용하는 방법에 따라 다릅니다. 그러나 다음 팁은 필요에 맞는 최적의 레이아웃을 찾는 데 도움이 될 수 있습니다.

- 예를 들어 특정 파티션이 특정 디스크에 있어야 하는 경우 특정 요구 사항이 있는 파티션을 만듭니다.
- 중요한 데이터가 포함될 수 있는 파티션 및 볼륨을 암호화하는 것이 좋습니다. 암호화는 권한이 없는 사용자가 물리적 스토리지 장치에 액세스할 수 있더라도 파티션의 데이터에 액세스하지 못하도록 합니다. 대부분의 경우 사용자 데이터가 포함된 **/home** 파티션을 암호화해야 합니다.
- 경우에 따라 **/** , **/boot** 및 **/ home** 이외의 디렉토리에 대해 별도의 마운트 지점을 만드는 것이 유용할 수 있습니다. 예를 들어 **MySQL** 데이터베이스를 실행하는 서버에서 **/var/lib/mysql** 에 대한 별도의 마운트 지점을 사용하면 백업에서 복원하지 않고도 다시 설치 중에 데이터베이스를 보

존할 수 있습니다. 그러나 불필요한 별도의 마운트 지점이 있으면 스토리지 관리가 더 어려워집니다.

- 일부 특수 제한 사항은 파티션 레이아웃을 배치할 수 있는 특정 디렉터리에 적용됩니다. 특히 **/boot** 디렉토리는 항상 물리 파티션에 있어야 합니다(**LVM** 볼륨 제외).

- **Linux**를 처음 사용하는 경우 다양한 시스템 디렉토리 및 콘텐츠에 대한 정보는 **Linux Filesystem Hierarchy** 표준을 검토하십시오.

- 각 커널에는 약 **60MiB(initrd 34MiB, 11MiB vmlinuz, 5MiB System.map)**가 필요합니다.

- 복구 모드의 경우: **100MiB(initrd 76MiB, 11MiB vmlinuz, 5MiB** 시스템 맵**)**

- **kdump** 가 시스템에서 활성화되면 약 **40MiB(23MiB**의 다른 **initrd)**가 사용됩니다.

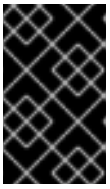
  대부분의 일반적인 사용 사례에 대해 **/boot** 에 대한 기본 파티션 크기 **1GiB**가 충분해야 합니다. 그러나 여러 커널 릴리스 또는 에라타 커널을 유지하려는 경우 이 파티션의 크기를 늘리는 것이 좋습니다.

- **/var** 디렉터리에는 **Apache** 웹 서버를 포함한 여러 애플리케이션에 대한 콘텐츠가 있으며, **YUM** 패키지 관리자가 다운로드한 패키지 업데이트를 임시로 저장하는 데 사용됩니다. **/var** 을 포함하는 파티션 또는 볼륨에 **3GiB** 이상 있는지 확인합니다.

- **/usr** 디렉토리는 일반적인 **Red Hat Enterprise Linux** 설치에 대부분의 소프트웨어를 보유하고 있습니다. 따라서 이 디렉터리를 포함하는 파티션 또는 볼륨은 최소 설치의 경우 **5GiB** 이상, 그래픽 환경이 있는 설치의 경우 **10GiB** 이상이어야 합니다.

- **/usr** 또는 **/var** 이 나머지 루트 볼륨과 별도로 분할된 경우 이러한 디렉토리에 부팅 중요한 구성 요소가 포함되어 있기 때문에 부팅 프로세스가 훨씬 더 복잡해집니다. 이러한 디렉터리가 **iSCSI** 드라이브 또는 **FCoE** 위치에 배치되는 경우와 같은 일부 상황에서는 시스템을 부팅할 수 없거나 전원을 끄거나 재부팅할 때 장치가 사용 중 오류가 발생할 수 있습니다.

  이 제한은 **/usr** 또는 **/var** 에만 적용되며, 해당 디렉터리에는 적용되지 않습니다. 예를 들어 **/var/www** 에 대한 별도의 파티션은 문제 없이 작동합니다.



중요

일부 보안 정책은 관리가 더 복잡하더라도 /usr 및 /var 를 분리해야 합니다.

- LVM 볼륨 그룹에 공간 일부를 할당되지 않은 상태로 두는 것이 좋습니다. 할당되지 않은 이 공간을 사용하면 공간 요구 사항이 변경되지만 다른 볼륨에서 데이터를 제거하고자 하는 경우 유연성이 제공됩니다. 파티션의 LVM 씬 프로비저닝 장치 유형을 선택하여 볼륨에서 사용되지 않은 공간을 자동으로 처리할 수도 있습니다.

- XFS 파일 시스템의 크기는 줄어들 수 없습니다. 이 파일 시스템으로 파티션이나 볼륨을 작게 만들어야 하는 경우 데이터를 백업하고 파일 시스템을 삭제한 다음 그 자리에 새로 더 작은 하나를 생성해야 합니다. 따라서 나중에 파티션 레이아웃을 변경하려는 경우 대신 ext4 파일 시스템을 사용해야 합니다.

- 하드 드라이브를 추가하거나 설치 후 가상 머신 하드 드라이브를 확장하여 스토리지를 확장하려는 경우 LVM(Logical Volume Management)을 사용합니다. LVM을 사용하면 새 드라이브에 물리 볼륨을 만든 다음 적합한 볼륨 그룹과 논리 볼륨에 할당할 수 있습니다. 예를 들어 시스템의 /home (또는 논리 볼륨에 상주하는 다른 디렉터리)을 쉽게 확장할 수 있습니다.

- 시스템의 펌웨어, 부팅 드라이브 크기 및 부팅 드라이브 디스크 레이블에 따라 BIOS 부팅 파티션 또는 EFI 시스템 파티션을 생성해야 할 수 있습니다. 시스템에 필요하지 않은 경우 그래픽 설치에서 BIOS Boot 또는 EFI 시스템 파티션을 생성할 수 없습니다. 이 경우 메뉴에서 숨겨집니다.

- 설치 후 스토리지 구성을 변경해야 하는 경우 Red Hat Enterprise Linux 리포지토리는 다음과 같은 여러 가지 도구를 제공합니다. 명령줄 툴을 선호하는 경우 system-storage-manager 를 사용해 보십시오.

추가 리소스

- IBM Z, LinuxONE 및 PAES 암호에서 dm-crypt를 사용하는 방법

C.6. 지원되는 하드웨어 스토리지

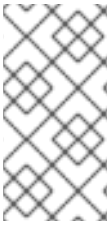
스토리지 기술이 구성되는 방법과 Red Hat Enterprise Linux의 주요 버전 간에 지원되는 방식을 이해하는 것이 중요합니다.

하드웨어 RAID

컴퓨터의 메인 보드 또는 연결된 컨트롤러 카드에서 제공하는 모든 **RAID** 기능은 설치 프로세스를 시작하기 전에 구성해야 합니다. 각 활성 **RAID** 어레이는 **Red Hat Enterprise Linux** 내에서 하나의 드라이브로 나타납니다.

소프트웨어 RAID

하드 드라이브가 **2**개 이상인 시스템에서는 **Red Hat Enterprise Linux** 설치 프로그램을 사용하여 여러 드라이브를 **Linux** 소프트웨어 **RAID** 배열로 작동할 수 있습니다. 소프트웨어 **RAID** 배열에서는 **RAID** 기능이 전용 하드웨어가 아닌 운영 체제에 의해 제어됩니다.



참고

기존 **RAID** 배열의 멤버 장치가 모두 파티션되지 않은 디스크/드라이브인 경우 설치 프로그램은 배열을 디스크로 처리하고 배열을 제거할 방법이 없습니다.

USB 디스크

설치 후 외부 **USB** 스토리지를 연결하고 구성할 수 있습니다. 대부분의 장치는 커널에 의해 인식되지만 일부 장치는 인식되지 않을 수 있습니다. 설치 중에 이러한 디스크를 구성해야 하는 경우 잠재적인 문제를 방지하기 위해 연결을 끊습니다.

NVDIMM 장치

NVDIMM(Non-Volatile Dual In-line Memory Module) 장치를 스토리지로 사용하려면 다음 조건을 충족해야 합니다.

- **Red Hat Enterprise Linux** 버전은 **7.6** 이상입니다.
- 시스템의 아키텍처는 **Intel 64** 또는 **AMD64**입니다.
- 장치는 섹터 모드로 구성됩니다. **Anaconda**는 **NVDIMM** 장치를 이 모드로 재구성할 수 있습니다.
- **nd_pmem** 드라이버에서 장치를 지원해야 합니다.

다음과 같은 추가 조건에서 **NVDIMM** 장치에서 부팅할 수 있습니다.

- *시스템은 **UEFI**를 사용합니다.*

- *장치는 시스템에서 사용 가능한 펌웨어 또는 **UEFI** 드라이버에서 지원해야 합니다. **UEFI** 드라이버는 장치 자체의 옵션 롬에서 로드될 수 있습니다.*

- *장치는 네임스페이스에서 사용할 수 있도록 설정해야 합니다.*

*부팅 중에 **NVDIMM** 장치의 높은 성능을 활용하려면 장치에 **/boot** 및 **/boot/efi** 디렉토리를 배치합니다.*

참고

***NVDIMM** 장치의 **XIP**(런타임) 기능은 부팅 중에 지원되지 않으며 커널이 기존 메모리에 로드됩니다.*

***Intel BIOS RAID Sets** 고려 사항*

***Red Hat Enterprise Linux**는 **Intel BIOS RAID** 세트에 **mdraid** 를 사용합니다. 이러한 세트는 부팅 프로세스 중에 자동으로 감지되며 장치 노드 경로는 여러 부팅 프로세스에서 변경될 수 있습니다. 장치 노드 경로(예: **/dev/sda**)를 파일 시스템 레이블 또는 장치 **UUID**로 교체하는 것이 좋습니다. **blkid** 명령을 사용하여 파일 시스템 레이블 및 장치 **UUID**를 찾을 수 있습니다.*