Red Hat Enterprise Linux 9

기본 시스템 설정 구성

Red Hat Enterprise Linux 9의 기본 시스템 설정 구성 가이드

Last Updated: 2022-08-21

Red Hat Enterprise Linux 9 기본 시스템 설정 구성

Red Hat Enterprise Linux 9의 기본 시스템 설정 구성 가이드

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

초록

이 문서에서는 Red Hat Enterprise Linux 9의 시스템 관리에 대한 기본 사항을 설명합니다. 제목은 운영 체제가 성공적으로 설치된 직후 시스템 관리자가 수행해야 하는 기본 작업, 서비스 관리에 systemd를 사용한 소프트웨어 설치, 사용자, 그룹 및 파일 권한 관리, chrony를 사용하여 NTP 등을 구성하는 작업에 중점을 두고 있습니다.

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

문서 개선을 위한 의견을 보내 주십시오. 문서를 개선할 수 있는 방법에 관해 알려주십시오.

- 특정 문구에 대한 간단한 의견 작성 방법은 다음과 같습니다.
  1. 문서가 *Multi-page HTML* 형식으로 표시되는지 확인합니다. 또한 문서 오른쪽 상단에 **피드백** 버튼이 있는지 확인합니다.
  2. 마우스 커서를 사용하여 주석 처리하려는 텍스트 부분을 강조 표시합니다.
  3. 강조 표시된 텍스트 아래에 표시되는 **피드백 추가** 팝업을 클릭합니다.
  4. 표시된 지침을 따릅니다.
- Bugzilla를 통해 피드백을 제출하려면 새 티켓을 생성하십시오.
  1. Bugzilla 웹 사이트로 이동하십시오.
  2. 구성 요소로 **문서**를 사용합니다.
  3. **설명** 필드에 문서 개선을 위한 제안 사항을 기입하십시오. 관련된 문서의 해당 부분 링크를 알려주십시오.
  4. **버그 제출**을 클릭합니다.

1장. RHEL 시스템 역할 시작하기

이 섹션에서는 RHEL 시스템 역할에 대해 설명합니다. 또한 Ansible 플레이북을 통해 특정 역할을 적용하여 다양한 시스템 관리 작업을 수행하는 방법을 설명합니다.

1.1. RHEL 시스템 역할 소개

RHEL 시스템 역할은 Ansible 역할 및 모듈의 컬렉션입니다. RHEL 시스템 역할은 여러 RHEL 시스템을 원격으로 관리하는 구성 인터페이스를 제공합니다. 이 인터페이스를 사용하면 여러 버전의 RHEL에서 시스템 구성을 관리하고 새로운 주요 릴리스를 채택할 수 있습니다.

Red Hat Enterprise Linux 9에서 인터페이스는 현재 다음 역할로 구성됩니다.

- 인증서 문제 및 업데이트
- 커널 설정
- 지표
- 네트워크 Bound 디스크 암호화 클라이언트 및 네트워크 Bound Disk Encryption 서버
- 네트워킹
- nsb
- SSH 클라이언트
- SSH 서버
- 시스템 전체 암호화 정책
- 터미널 세션 기록

이러한 모든 역할은 **AppStream** 리포지토리에서 사용할 수 있는 **rhel-system-roles** 패키지에서 제공합니다.

추가 리소스

- Red Hat Enterprise Linux (RHEL) 시스템 역할
- **/usr/share/doc/rhel-system-roles/** 디렉터리에 있는 문서 [1]

1.2. RHEL 시스템 역할 용어

이 문서에서 다음 용어를 찾을 수 있습니다.

Ansible 플레이북

플레이북은 Ansible의 구성, 배포 및 오케스트레이션 언어입니다. 원격 시스템을 적용하려는 정책이나 일반 IT 프로세스에서 일련의 단계를 설명할 수 있습니다.

제어 노드

Ansible이 설치된 모든 시스템. 모든 제어 노드에서 /usr/bin/ansible 또는 /usr/bin/ansible-playbook을 호출하는 명령과 플레이북을 실행할 수 있습니다. Python이 설치된 모든 컴퓨터를 제어 노드로 사용할 수 있습니다. 즉 랩탑, 공유 데스크탑 및 서버는 모두 Ansible을 실행할 수 있습니다. 그러나 Windows 머신을 제어 노드로 사용할 수 없습니다. 여러 컨트롤 노드를 사용할 수 있습니다.

인벤토리

관리형 노드 목록입니다. 인벤토리 파일을 "hostfile"이라고도 합니다. 인벤토리는 각 관리 노드에 대해 IP 주소와 같은 정보를 지정할 수 있습니다. 인벤토리는 보다 쉽게 스케일링할 수 있도록 관리형 노드를 구성하고, 그룹을 생성 및 중첩할 수 있습니다. 인벤토리에 대한 자세한 내용은 Working with Inventory 섹션을 참조하십시오.

관리형 노드

Ansible로 관리하는 네트워크 장치, 서버 또는 둘 다입니다. 관리형 노드를 "hosts"라고도 합니다. Ansible은 관리형 노드에 설치되지 않습니다.

1.3. 역할 적용

다음 절차에서는 특정 역할을 적용하는 방법을 설명합니다.

사전 요구 사항

- **rhel-system-roles** 패키지가 제어 노드로 사용하려는 시스템에 설치되어 있는지 확인합니다.

```
# dnf install rhel-system-roles
```

1. Ansible Core 패키지를 설치합니다.

```
# dnf install ansible-core
```

Ansible Core 패키지는 **ansible-playbook** CLI, Ansible Vault 기능 및 RHEL Ansible 콘텐츠에 필요한 기본 모듈 및 필터를 제공합니다.

- Ansible 인벤토리를 생성할 수 있는지 확인합니다.
인벤토리는 Ansible 플레이북에서 사용하는 호스트, 호스트 그룹 및 일부 구성 매개변수를 나타냅니다.

  플레이북은 일반적으로 사람이 읽을 수 있으며 **ini**, yaml**, json** 및 기타 파일 형식으로 정의됩니다.

- Ansible 플레이북을 생성할 수 있는지 확인합니다.
플레이북은 Ansible의 구성, 배포 및 오케스트레이션 언어를 나타냅니다. 플레이북을 사용하면 원격 시스템의 구성을 선언 및 관리하고 여러 원격 시스템을 배포하거나 수동 순서가 지정된 프로세스의 단계를 오케스트레이션할 수 있습니다.

  플레이북은 하나 이상의 **플레이** 목록입니다. 모든 **플레이** 에는 Ansible 변수, 작업 또는 역할이 포함될 수 있습니다.

  플레이북은 사람이 읽을 수 있으며 yaml 형식으로 **정의됩니다**.

절차

1. 관리하려는 호스트 및 그룹을 포함하는 필수 Ansible 인벤토리를 생성합니다. 다음과 같습니다. inventory.ini라는 호스트 그룹의 **inventory.ini** 파일을 사용하는 예입니다.

```
[webservers]
host1
host2
host3
```

2.

필요한 역할을 포함하여 **Ansible** 플레이북을 생성합니다. 다음 예는 플레이북에 **roles:** 옵션을 통해 역할을 사용하는 방법을 보여줍니다.

다음 예제에서는 지정된 플레이에 대해 **roles:** 옵션을 통해 역할을 사용하는 방법을 보여줍니다.

```
---
- hosts: webservers
  roles:

    - rhel-system-roles.network
    - rhel-system-roles.postfix
```

참고

모든 역할에는 **README** 파일이 포함되어 있으며, 이 파일은 역할 및 지원되는 매개 변수 값을 사용하는 방법을 설명합니다. 역할의 문서 디렉터리에서 특정 역할에 대한 예제 플레이북을 찾을 수도 있습니다. 이러한 문서 디렉터리는 기본적으로 **rhel-system-roles** 패키지와 함께 제공되며 다음 위치에서 확인할 수 있습니다.

```
/usr/share/doc/rhel-system-roles/SUBSYSTEM/
```

SUBSYSTEM 을 **postfix,metrics,network,tlog** 또는 **ssh** 와 같은 필수 역할의 이름으로 바꿉니다.

3.

특정 호스트에서 플레이북을 실행하려면 다음 중 하나를 수행해야 합니다.

- **hosts: host1[,host2,…**을 사용하도록 플레이북을 편집합니다.], 또는 호스트**: all, and execute the command:**

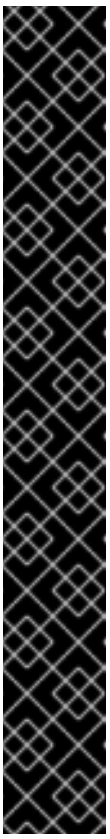
```
# ansible-playbook name.of.the.playbook
```

- 인벤토리를 편집하여 사용하려는 호스트가 그룹에 정의되어 있는지 확인하고 명령을 실행합니다.

```
# ansible-playbook -i name.of.the.inventory name.of.the.playbook
```

- **ansible-playbook** 명령을 실행할 때 모든 호스트를 지정합니다.

```
# ansible-playbook -i host1,host2,... name.of.the.playbook
```



중요

-i 플래그는 사용 가능한 모든 호스트의 인벤토리를 지정합니다. 대상 호스트가 여러 개 있지만 플레이북을 실행할 호스트를 선택하려면 플레이북에서 변수를 추가하여 호스트를 선택할 수 있습니다. 예를 들면 다음과 같습니다.

```
Ansible Playbook | example-playbook.yml:


- hosts: "{{ target_host }}"
  roles:
    - rhel-system-roles.network
    - rhel-system-roles.postfix
```

Playbook 실행 명령:

```
# ansible-playbook -i host1,..hostn -e target_host=host5 example-
playbook.yml
```

추가 리소스

- Ansible Playbook
- Ansible 플레이북에서 역할 사용
- Ansible 플레이북의 예
- 인벤토리를 만들고 작업하는 방법은 무엇입니까?
- ansible-playbook 툴

1.4. 추가 리소스

- Red Hat Enterprise Linux (RHEL) 시스템 역할
- RHEL 시스템 역할을 사용하여 로컬 스토리지 관리
- RHEL 시스템 역할을 사용하여 여러 시스템에 동일한 SELinux 구성 배포

---

[1] 이 문서는 **rhel-system-roles** 패키지를 사용하여 자동으로 설치됩니다.

2장. 기본 환경 설정 변경

기본 환경 설정 구성은 설치 프로세스의 일부입니다. 다음 섹션에서는 나중에 변경할 때 안내합니다. 환경의 기본 구성은 다음과 같습니다.

- 날짜 및 시간
- 시스템 로케일
- 키보드 레이아웃
- 언어

2.1. 날짜 및 시간 구성

정확한 유지 관리는 여러 가지 이유로 중요합니다. **Red Hat Enterprise Linux**에서 시간 유지는 **NTP** 프로토콜에 의해 확인되며, 이는 사용자 공간에서 실행되는 데몬에 의해 구현됩니다. 사용자 공간 데몬은 커널에서 실행되는 시스템 클럭을 업데이트합니다. 시스템 클럭은 다양한 클럭 소스를 사용하여 시간을 유지할 수 있습니다.

Red Hat Enterprise Linux 8 이상 버전은 **chronyd** 데몬을 사용하여 **NTP** 를 구현합니다.**chronyd** 는 **chrony** 패키지에서 사용할 수 있습니다. 자세한 내용은 **chrony Suite**를 사용하여 **NTP** 설정을 참조하십시오.

2.1.1. 현재 날짜 및 시간 표시

현재 날짜 및 시간을 표시하려면 다음 단계 중 하나를 사용합니다.

절차

1. **date** 명령을 입력합니다.

```
$ date
Mon Mar 30 16:02:59 CEST 2020
```

2. 자세한 내용을 보려면 **timedatectl** 명령을 사용하십시오.

```
$ timedatectl
Local time: Mon 2020-03-30 16:04:42 CEST
Universal time: Mon 2020-03-30 14:04:42 UTC
  RTC time: Mon 2020-03-30 14:04:41
 Time zone: Europe/Prague (CEST, +0200)
System clock synchronized: yes
NTP service: active
RTC in local TZ: no
```

추가 리소스

- 웹 콘솔을 사용하여 시간 설정 구성
- **man date(1)** 및 **man timedatectl(1)**

2.2. 시스템 로케일 구성

시스템 전체 로케일 설정은 **systemd** 데몬에서 초기 부팅 시 읽는 **/etc/locale.conf** 파일에 저장됩니다. 모든 서비스 또는 사용자는 개별 프로그램 또는 개별 사용자가 재정의하지 않는 한 **/etc/locale.conf** 에 구성된 로케일 설정을 상속합니다.

이 섹션에서는 시스템 로케일을 관리하는 방법에 대해 설명합니다.

절차

- 사용 가능한 시스템 로케일 설정을 나열하려면 다음을 수행합니다.

```
$ localectl list-locales
C.utf8
aa_DJ
aa_DJ.iso88591
aa_DJ.utf8
...
```

- 시스템 로케일 설정의 현재 상태를 표시하려면 다음을 수행합니다.

```
$ localectl status
```

- 기본 시스템 로케일 설정을 설정하거나 변경하려면 **root** 사용자로 **localectl set-locale** 하위 명령을 사용합니다. 예를 들면 다음과 같습니다.

```
# localectl set-locale LANG=en_US
```

추가 리소스

- **man localectl(1), man locale(7)** 및 **man locale.conf(5)**

2.3. 키보드 레이아웃 구성

키보드 레이아웃 설정은 텍스트 콘솔 및 그래픽 사용자 인터페이스에서 사용되는 레이아웃을 제어합니다.

절차

- 사용 가능한 키맵을 나열하려면 다음을 수행합니다.

```
$ localectl list-keymaps
ANSI-dvorak
al
al-plisi
amiga-de
amiga-us
...
```

- **keymaps** 설정의 현재 상태를 표시하려면 다음을 수행합니다.

```
$ localectl status
...
VC Keymap: us
...
```

- 기본 시스템 키 맵을 설정하거나 변경하려면 다음을 수행합니다. 예를 들면 다음과 같습니다.

```
# localectl set-keymap us
```

추가 리소스

- man localectl(1), man locale(7) 및 man locale.conf(5)

2.4. 데스크탑 GUI를 사용하여 언어 변경

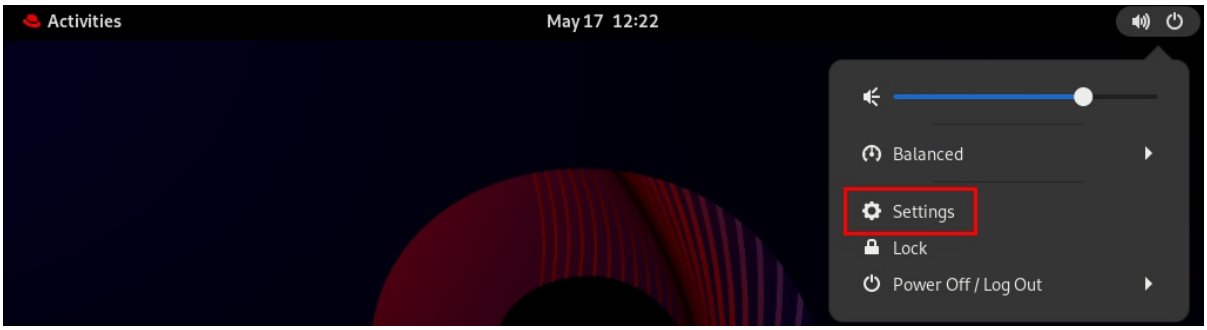
이 섹션에서는 데스크톱 GUI를 사용하여 시스템 언어를 변경하는 방법을 설명합니다.

사전 요구 사항

- 필수 언어 패키지가 시스템에 설치되어 있습니다.

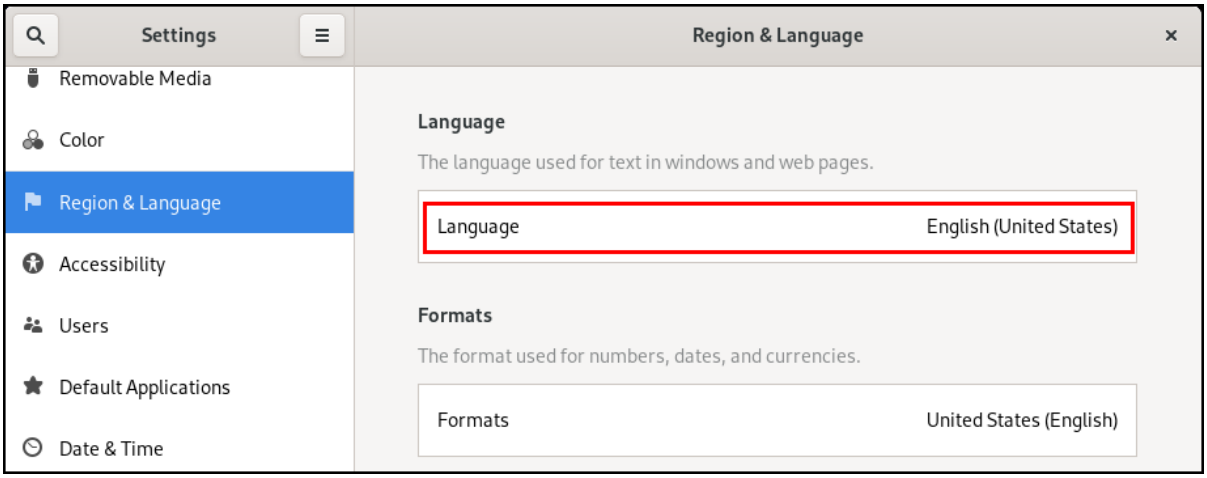
절차

1. 아이콘을 클릭하여 시스템 메뉴에서 GNOME Control Center 를 엽니다.

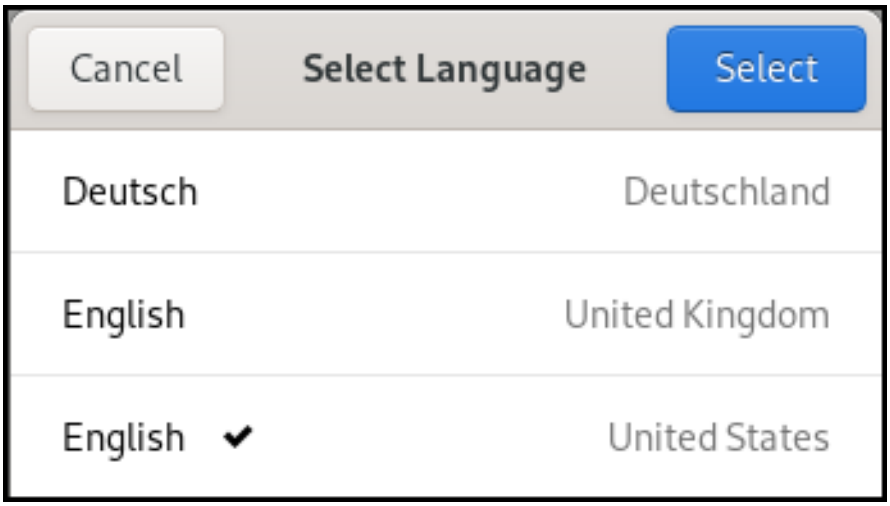


2. GNOME Control Center (GNOME 제어 센터)의 왼쪽 수직 표시줄에서 Region & Language 를 선택합니다.

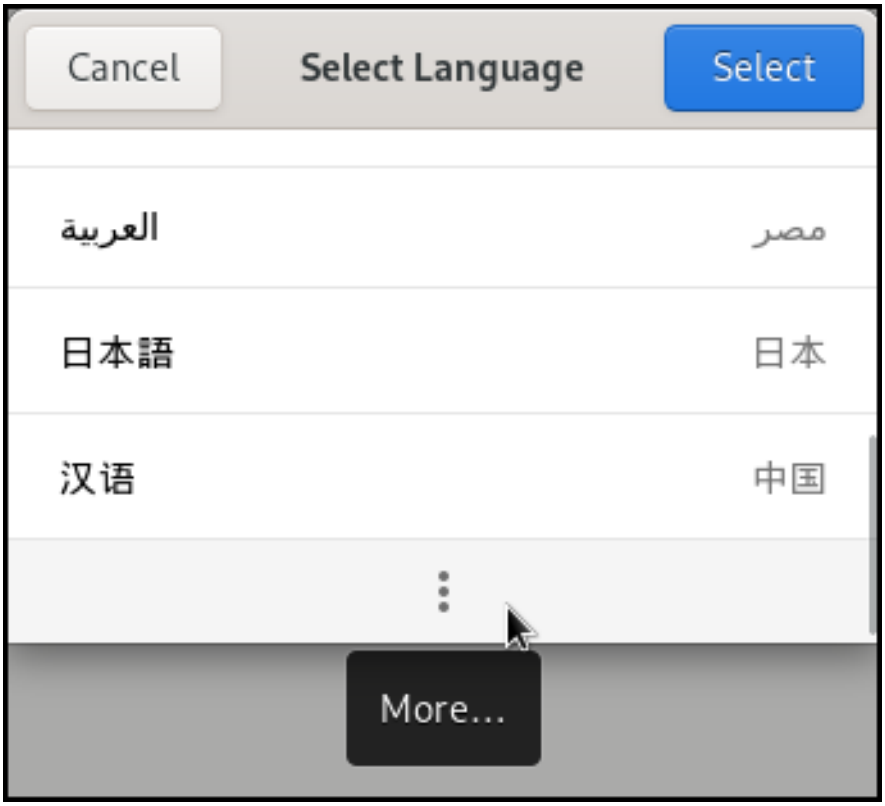
3. Language 메뉴를 클릭합니다.



4. 메뉴에서 필요한 지역 및 언어를 선택합니다.

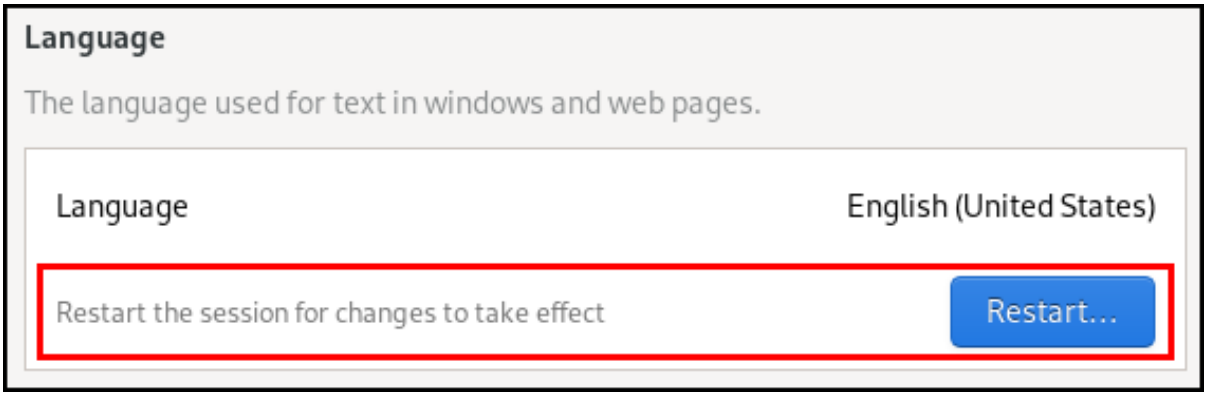


지역 및 언어가 나열되지 않은 경우 아래로 스크롤하여 **More** 를 클릭하여 사용 가능한 지역 및 언어를 선택합니다.



5. **Done(완료)을 클릭합니다.**

6. **변경 사항을 적용하려면 Restart 를 클릭합니다.**



참고

일부 애플리케이션은 특정 언어를 지원하지 않습니다. 선택한 언어로 번역할 수 없는 애플리케이션의 텍스트는 미국 영어로 유지됩니다.

추가 리소스

- GNOME에서 애플리케이션 시작

2.5. 추가 리소스

- 표준 RHEL 9 설치 수행

3장. 네트워크 액세스 구성 및 관리

이 섹션에서는 **Red Hat Enterprise Linux**에서 이더넷 연결을 추가하는 방법에 대한 다양한 옵션에 대해 설명합니다.

3.1. 그래픽 설치 모드에서 네트워크 및 호스트 이름 구성

이 절차의 단계에 따라 네트워크 및 호스트 이름을 구성합니다.

절차

1. 설치 요약 창에서 네트워크 및 호스트 이름을 클릭합니다.

2. 왼쪽 창의 목록에서 인터페이스를 선택합니다. 자세한 내용은 오른쪽 창에 표시됩니다.

참고

- 영구적인 이름으로 네트워크 장치를 식별하는 데 사용되는 네트워크 장치 이름 지정 표준 유형은 여러 가지가 있습니다(예: **em1** 및 **wl3sp0**). 이러한 표준에 대한 자세한 내용은 *네트워킹 구성 및 관리를* 참조하십시오.

3. 선택한 인터페이스를 활성화하거나 비활성화하려면 **ON/OFF** 스위치를 전환합니다.

참고

설치 프로그램은 로컬 액세스 가능한 인터페이스를 자동으로 감지하므로 수동으로 추가하거나 제거할 수 없습니다.

4. **+** 를 클릭하여 다음 중 하나일 수 있는 가상 네트워크 인터페이스를 추가합니다. 팀(폐기됨), 본딩, 브리지 또는 **VLAN.**

5. 가상 인터페이스를 제거하려면 **-** 를 클릭합니다.

6. **Configure** (구성)를 클릭하여 기존 인터페이스(가상 및 물리적)의 **IP** 주소, **DNS** 서버 또는 라

우팅 구성과 같은 설정을 변경합니다.

7. 시스템의 호스트 이름을 호스트 이름 필드에 입력합니다.

참고

- 호스트 이름은 ***hostname.domainname*** 형식의 정규화된 도메인 이름 **(FQDN)**이거나 도메인 이름이 없는 짧은 호스트 이름일 수 있습니다. 대부분의 네트워크에는 연결된 시스템을 도메인 이름으로 자동으로 공급하는 **DHCP(Dynamic Host Configuration Protocol)** 서비스가 있습니다. **DHCP** 서비스에서 도메인 이름을 이 시스템에 할당할 수 있도록 하려면 짧은 호스트 이름만 지정합니다. **localhost** 값은 대상 시스템의 특정 정적 호스트 이름이 구성되지 않음을 의미하며, 설치된 시스템의 실제 호스트 이름은 네트워크 구성(예: **DHCP** 또는 **DNS**를 사용하는 **NetworkManager**)에서 구성됩니다.
- 호스트 이름은 **alpha-numeric** 문자와 **-** 또는 **.** 만 포함할 수 있습니다. 호스트 이름은 **-** 및 로 시작하거나 종료할 수 없습니다 **.**

8. **Apply(**적용 **)**를 클릭하여 설치 프로그램 환경에 호스트 이름을 적용합니다.

9. 또는 네트워크 및 호스트 이름 창에서 무선 옵션을 선택할 수 있습니다. 오른쪽 창에서 네트워크 선택을 클릭하여**eo** 연결을 선택하고 필요한 경우 암호를 입력한 다음 완료 를 클릭합니다.

추가 리소스

- 고급 RHEL 9 설치 수행

3.2. NMCLI를 사용하여 정적 이더넷 연결 구성

다음 절차에서는 **nmcli** 유틸리티를 사용하여 다음 설정으로 이더넷 연결을 추가하는 방법을 설명합니다.

- 정적 **IPv4** 주소 **- 192.0.2.1** 에 **/24** 서브넷 마스크
- 정적 **IPv6** 주소 **- 2001:db8:1::1** 에 **/64** 서브넷 마스크

- IPv4 기본 게이트웨이 - 192.0.2.254
- IPv6 기본 게이트웨이 - 2001:db8:1::fffe
- IPv4 DNS 서버 - 192.0.2.200
- IPv6 DNS 서버 - 2001:db8:1::ffbb
- DNS 검색 도메인 - example.com

절차

1. 이더넷 연결을 위해 새 NetworkManager 연결 프로필을 추가합니다.

```
# nmcli connection add con-name Example-Connection ifname enp7s0 type ethernet
```

추가 단계는 생성한 Example-Connection 연결 프로필을 수정합니다.

2. IPv4 주소를 설정합니다.

```
# nmcli connection modify Example-Connection ipv4.addresses 192.0.2.1/24
```

3. IPv6 주소를 설정합니다.

```
# nmcli connection modify Example-Connection ipv6.addresses 2001:db8:1::1/64
```

4. IPv4 및 IPv6 연결 방법을 manual 로 설정합니다.

```
# nmcli connection modify Example-Connection ipv4.method manual
# nmcli connection modify Example-Connection ipv6.method manual
```

5. IPv4 및 IPv6 기본 게이트웨이를 설정합니다.

```
# nmcli connection modify Example-Connection ipv4.gateway 192.0.2.254
# nmcli connection modify Example-Connection ipv6.gateway 2001:db8:1::fffe
```

6. IPv4 및 IPv6 DNS 서버 주소를 설정합니다.

```
# nmcli connection modify Example-Connection ipv4.dns "192.0.2.200"
# nmcli connection modify Example-Connection ipv6.dns "2001:db8:1::ffbb"
```

여러 DNS 서버를 설정하려면 공백으로 구분하여 따옴표로 묶습니다.

7. IPv4 및 IPv6 연결에 대한 DNS 검색 도메인을 설정합니다.

```
# nmcli connection modify Example-Connection ipv4.dns-search example.com
# nmcli connection modify Example-Connection ipv6.dns-search example.com
```

8. 연결 프로필을 활성화합니다.

```
# nmcli connection up Example-Connection
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/13)
```

검증 단계

1. 장치 및 연결의 상태를 표시합니다.

```
# nmcli device status
DEVICE     TYPE      STATE      CONNECTION
enp7s0     ethernet  connected  Example-Connection
```

2. 연결 프로필의 모든 설정을 표시하려면 다음을 수행합니다.

```
# nmcli connection show Example-Connection
connection.id:              Example-Connection
connection.uuid:            b6cdfa1c-e4ad-46e5-af8b-a75f06b79f76
connection.stable-id:       --
connection.type:            802-3-ethernet
connection.interface-name:  enp7s0
...
```

3.

ping 유틸리티를 사용하여 이 호스트에서 다른 호스트에 패킷을 보낼 수 있는지 확인합니다.

- 동일한 서브넷의 **IP** 주소를 **ping**합니다.

  IPv4의 경우:

  ```
  # ping 192.0.2.3
  ```

  IPv6의 경우:

  ```
  # ping 2001:db8:1::2
  ```

  명령이 실패하면 **IP** 및 서브넷 설정을 확인합니다.

- 원격 서브넷의 **IP** 주소를 **ping**합니다.

  IPv4의 경우:

  ```
  # ping 198.162.3.1
  ```

  IPv6의 경우:

  ```
  # ping 2001:db8:2::1
  ```

  - 명령이 실패하면 기본 게이트웨이를 **ping**하여 설정을 확인합니다.

    IPv4의 경우:

    ```
    # ping 192.0.2.254
    ```

    IPv6의 경우:

    ```
    # ping 2001:db8:1::fff3
    ```

4.

호스트 유틸리티를 사용하여 이름 확인이 작동하는지 확인합니다. 예를 들면 다음과 같습니다.

```
# host client.example.com
```

명령이 연결 시간이 초과 되거나 서버에 도달할 수 없는 것과 같은 오류를 반환하는 경우 **DNS** 설정을 확인합니다.

문제 해결 단계

1. 연결에 실패하거나 네트워크 인터페이스가 **up** 및 **down** 상태 간에 전환되는 경우:

   - 네트워크 케이블이 호스트와 스위치에 연결되어 있는지 확인합니다.
   - 이 호스트에만 링크 오류가 있는지 또는 서버가 연결된 동일한 스위치에 연결된 다른 호스트에도 연결 실패가 있는지 확인합니다.
   - 네트워크케이블 및 네트워크 인터페이스가 예상대로 작동하는지 확인합니다. 하드웨어 진단 단계를 수행하고 결함 **variables** 및 네트워크 인터페이스 카드를 교체합니다.
   - 디스크의 구성이 장치의 구성과 일치하지 않는 경우 **NetworkManager**를 시작하거나 다시 시작하면 장치의 구성을 반영하는 메모리 내 연결이 생성됩니다. 자세한 내용과 이 문제를 방지하는 방법에 대한 자세한 내용은 **NetworkManager** 서비스를 다시 시작한 후 NetworkManager 서비스를 다시 시작한 후 연결 중복을참조하십시오.

추가 리소스

- **nm-settings(5), nmcli** 및 **nmcli(1)** 도움말 페이지
- 기본 게이트웨이를 제공하기 위해 특정 프로필을 사용하지 않도록 **NetworkManager** 구성

3.3. NMTUI를 사용하여 연결 프로필 추가

nmtui 애플리케이션은 **NetworkManager**에 텍스트 사용자 인터페이스를 제공합니다. 다음 절차에서는 새 연결 프로필을 추가하는 방법을 설명합니다.

사전 요구 사항

- **NetworkManager-tui** 패키지가 설치됩니다.

절차

1. **NetworkManager** 텍스트 사용자 인터페이스 유틸리티를 시작합니다.

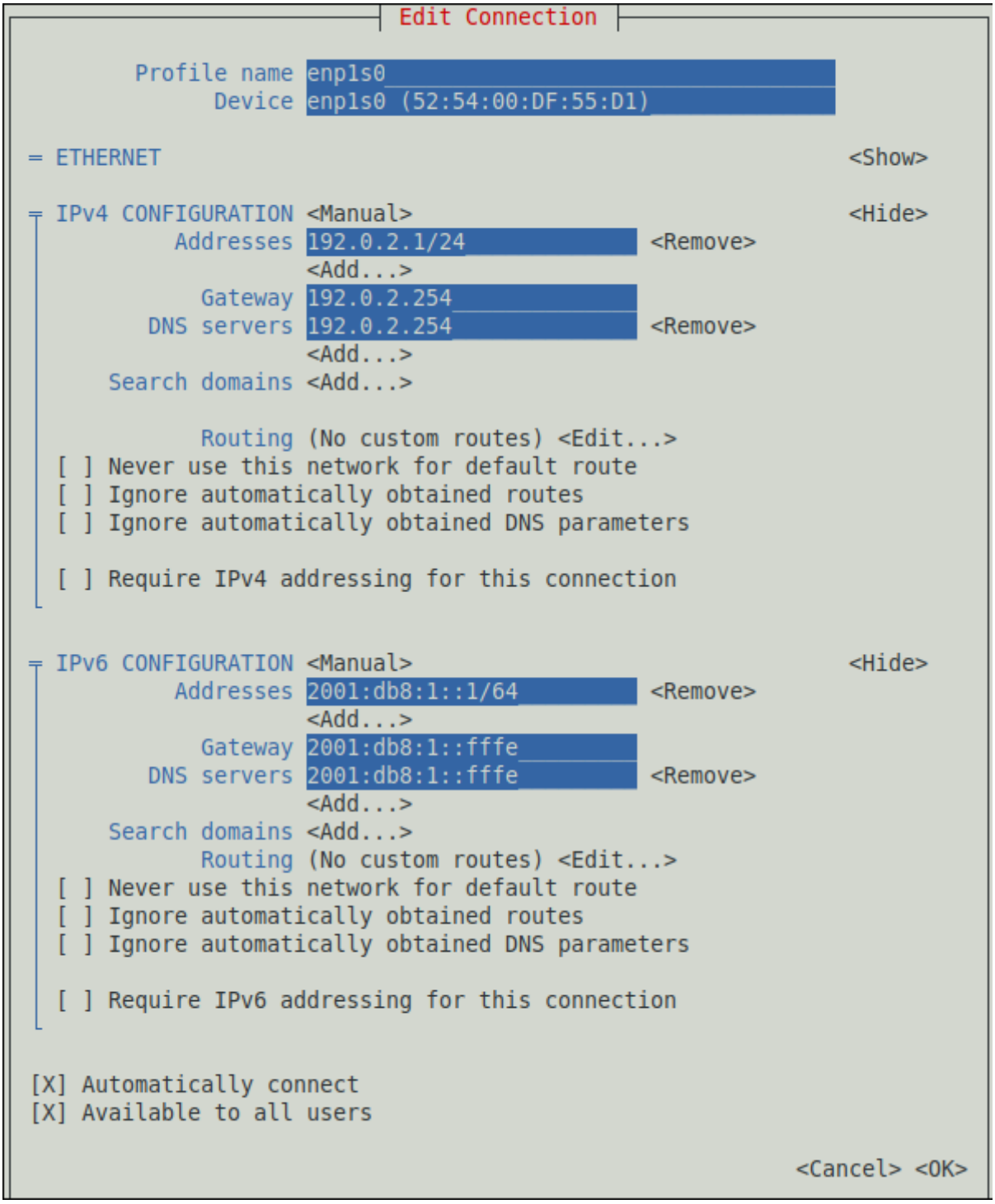
```
# nmtui
```

2. **Edit a connection** 메뉴 항목을 선택하고 **Enter** 키를 누릅니다.

3. 추가 버튼을 선택하고 **Enter** 를 누릅니다.

4. 이더넷 을 선택하고 **Enter** 를 누릅니다.

5. 연결 세부 정보를 사용하여 필드를 채웁니다.



6. 확인 을 선택하여 변경 내용을 저장합니다.**Select OK to save the changes.**

7. **Back** (뒤로)을 선택하여 메인 메뉴로 돌아갑니다.

8. **connecting a connection** 을 선택하고 **Enter** 키를 누릅니다.

9. 새 연결 항목을 선택하고 **Enter** 를 눌러 연결을 활성화합니다.

10. 메인 메뉴로 돌아가려면 **Back** 을 선택합니다.

11. **Quit** 을 선택합니다.

검증 단계

1. 장치 및 연결의 상태를 표시합니다.

```
# nmcli device status
DEVICE     TYPE     STATE      CONNECTION
enp1s0     ethernet  connected  Example-Connection
```

2. 연결 프로필의 모든 설정을 표시하려면 다음을 수행합니다.

```
# nmcli connection show Example-Connection
connection.id:            Example-Connection
connection.uuid:          b6cdfa1c-e4ad-46e5-af8b-a75f06b79f76
connection.stable-id:     --
connection.type:          802-3-ethernet
connection.interface-name:  enp1s0
...
```

디스크의 구성이 장치의 구성과 일치하지 않는 경우 **NetworkManager**를 시작하거나 다시 시작하면 장치의 구성을 반영하는 메모리 내 연결이 생성됩니다. 자세한 내용과 이 문제를 방지하는 방법에 대한 자세한 내용은 **NetworkManager** 서비스를 다시 시작한 후 **NetworkManager** 서비스를 다시 시작한 후 연결 중복 을 참조하십시오.

추가 리소스

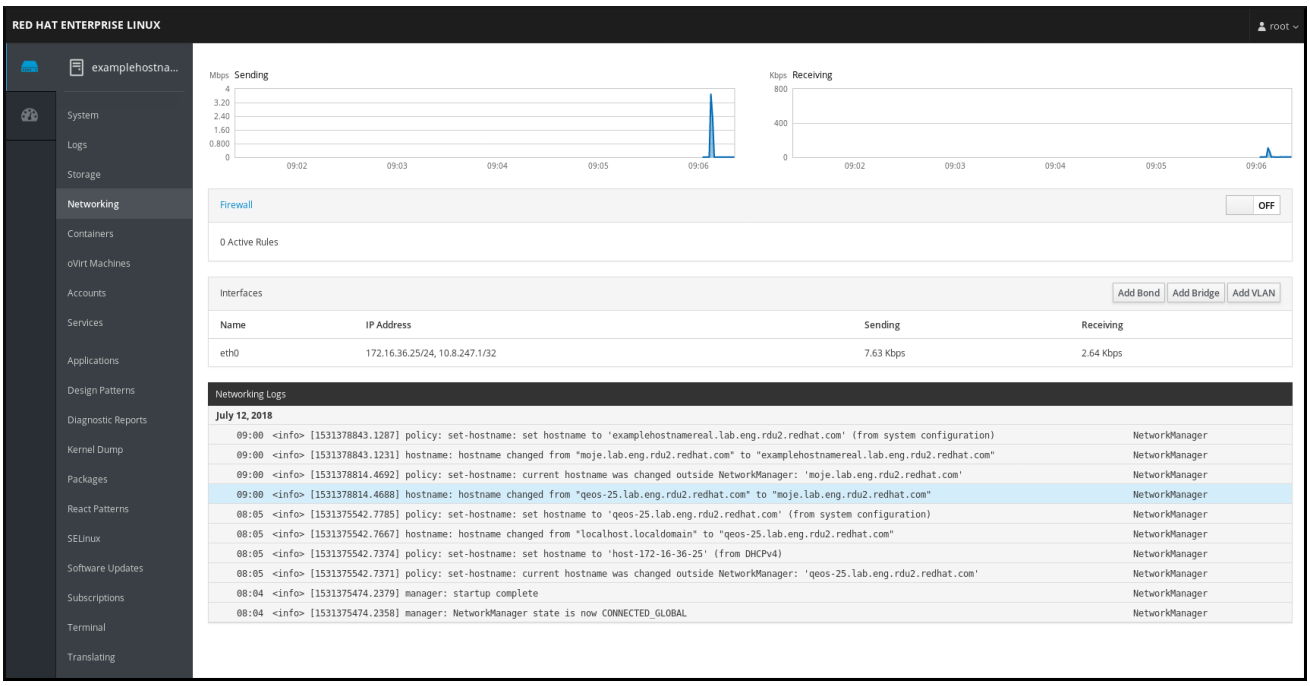
- 기본 네트워크 설정 테스트
- **nmtui(1)** 도움말 페이지

3.4. RHEL 웹 콘솔에서 네트워킹 관리

웹 콘솔에서 네트워킹 메뉴를 사용하면 다음을 수행할 수 있습니다.

- 현재 수신 및 전송된 패킷을 표시하려면 다음을 수행합니다.
- 사용 가능한 네트워크 인터페이스의 가장 중요한 특성 표시
- 네트워킹 로그의 콘텐츠를 표시하려면 다음을 수행합니다.
- 다양한 유형의 네트워크 인터페이스 추가 (bond, team, bridge, VLAN)

그림 3.1. RHEL 웹 콘솔에서 네트워킹 관리



3.5. RHEL 시스템 역할을 사용하여 네트워킹 관리

네트워크 역할을 사용하여 여러 대상 시스템에서 네트워킹 연결을 구성할 수 있습니다.

네트워크 역할을 사용하면 다음 유형의 인터페이스를 구성할 수 있습니다.

- 이더넷
- **Bridge**

- 본딩
- **VLAN**
- **MacVLAN**
- **InfiniBand**

각 호스트에 필요한 네트워킹 연결은 **network_connections** 변수 내에서 목록으로 제공됩니다.

주의

네트워크 역할은 **network _connections** 변수에 지정된 대로 정확하게 대상 시스템의 모든 연결 프로필을 업데이트하거나 생성합니다. 따라서 옵션이 시스템에만 존재하지만 **network _connections** 변수에 없는 경우 네트워크 역할은 지정된 프로필에서 옵션을 제거합니다.

다음 예제에서는 네트워크 역할을 적용하여 필수 매개 변수를 사용한 이더넷 연결이 있는지 확인하는 방법을 보여줍니다.

네트워크 역할을 적용하여 필요한 매개 변수를 사용하여 이더넷 연결을 설정하는 예제 Playbook

```
# SPDX-License-Identifier: BSD-3-Clause
---
- hosts: network-test
  vars:
    network_connections:

      # Create one ethernet profile and activate it.
      # The profile uses automatic IP addressing
      # and is tied to the interface by MAC address.
      - name: prod1
        state: up
        type: ethernet
```

```
      autoconnect: yes
      mac: "00:00:5e:00:53:00"
      mtu: 1450

roles:
  - rhel-system-roles.network
```

추가 리소스

- RHEL 시스템 역할 시작하기

3.6. 추가 리소스

- 네트워킹 구성 및 관리

4장. 시스템 등록 및 서브스크립션 관리

서브스크립션은 운영 체제 자체를 포함하여 **Red Hat Enterprise Linux**에 설치된 제품에 적용됩니다.

Red Hat Content Delivery Network 서브스크립션을 사용하여 다음을 추적할 수 있습니다.

- 등록된 시스템
- 시스템에 설치된 제품
- 설치된 제품에 연결된 서브스크립션

4.1. 설치 후 시스템 등록

이미 설치 프로세스 중에 등록하지 않은 경우 다음 절차를 사용하여 시스템을 등록합니다.

사전 요구 사항

- **Red Hat** 고객 포털의 유효한 사용자 계정.
- **Red Hat** 로그인 생성 페이지를 참조하십시오.
- **RHEL** 시스템의 활성 서브스크립션입니다.
- 설치 프로세스에 대한 자세한 내용은 표준 **RHEL 9** 설치 수행을 참조하십시오.

절차

1. 시스템을 한 단계로 등록하고 자동으로 등록합니다.

```
# subscription-manager register --username <username> --password <password> --auto-attach
Registering to: subscription.rhsm.redhat.com:443/subscription
```

```
The system has been registered with ID: 37to907c-ece6-49ea-9174-20b87ajk9ee7
The registered system name is: client1.idm.example.com
Installed Product Current Status:
Product Name: Red Hat Enterprise Linux for x86_64
Status:       Subscribed
```

이 명령은 **Red Hat Customer Portal** 사용자 이름과 암호를 입력하라는 메시지를 표시합니다.

등록 프로세스가 실패하면 시스템을 특정 풀로 등록할 수 있습니다. 이 작업을 수행하는 방법에 대한 지침은 다음 단계를 진행합니다.

a. 필요한 서브스크립션의 풀 **ID**를 확인합니다.

```
# subscription-manager list --available
```

이 명령은 **Red Hat** 계정에 사용 가능한 모든 서브스크립션을 표시합니다. 모든 서브스크립션에 대해 풀 **ID**를 포함하여 다양한 특성이 표시됩니다.

b. ***pool_id*** 를 이전 단계에서 확인한 풀 **ID**로 교체하여 시스템에 적절한 서브스크립션을 연결합니다.

```
# subscription-manager attach --pool=pool_id
```

참고

Red Hat Insights에 시스템을 등록하려면 **rhc connect** 유틸리티를 사용할 수 있습니다. **Red Hat** 커넥터 설정을 참조하십시오.

추가 리소스

- 고객 포털의 자동 연결 이해
- 고객 포털에서 수동 등록 및 서브스크립션 이해

4.2. 웹 콘솔에서 자격 증명을 사용하여 서브스크립션 등록

다음 단계를 사용하여 RHEL 웹 콘솔을 사용하여 계정 자격 증명으로 새로 설치된 Red Hat Enterprise Linux를 등록합니다.

사전 요구 사항

- Red Hat 고객 포털에서 유효한 사용자 계정.

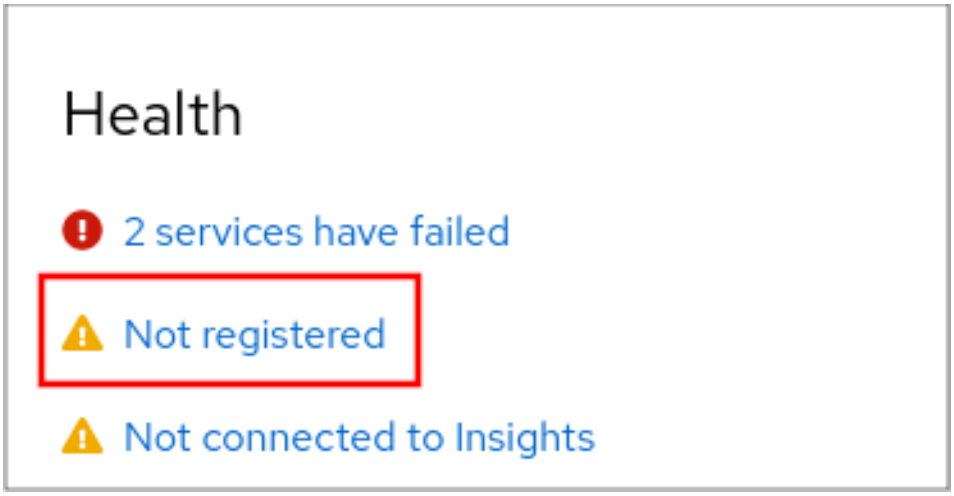
  Red Hat 로그인 생성 페이지를 참조하십시오.

- RHEL 시스템용 활성 서브스크립션.

절차

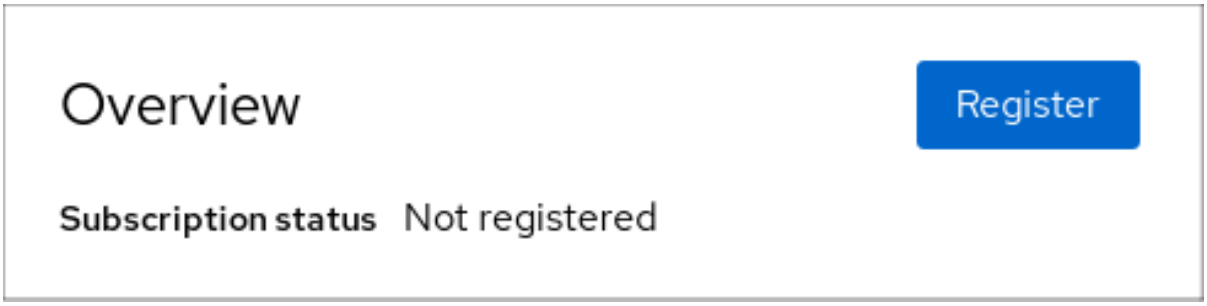
1. RHEL 웹 콘솔에 로그인합니다. 자세한 내용은 웹 콘솔에 로그인을 참조하십시오.

2. 개요 페이지의 Health filed에서 Not registered 경고를 클릭하거나 메인 메뉴에서 Subscriptions (서브스크립션)를 클릭하여 서브스크립션 정보를 페이지로 이동합니다.

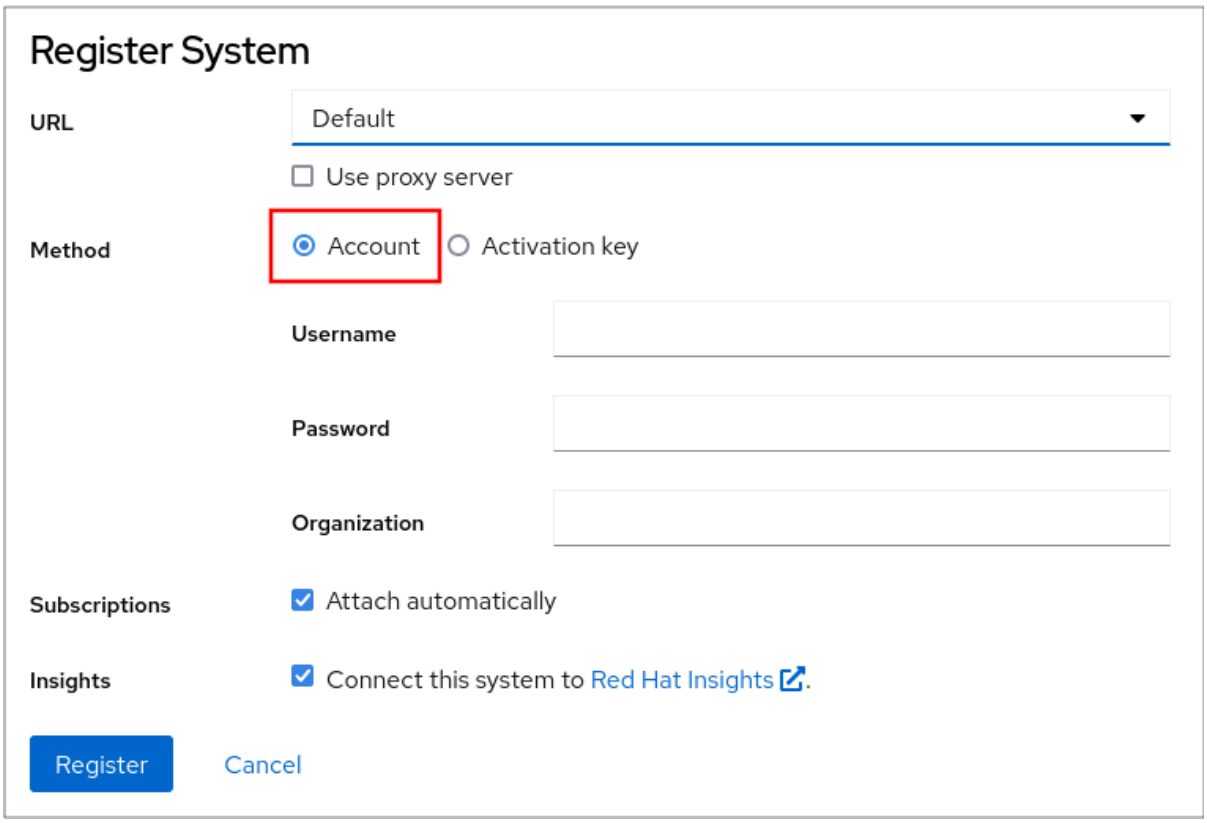


.

3. Overview filed에서 Register 를 클릭합니다.



4. 시스템 등록 대화 상자에서 계정 자격 증명을 사용하여 등록할 항목을 선택합니다.**In the Register system dialog box, select that you want to register using your account credentials.**



5. 사용자 이름을 입력합니다.

6. 암호를 입력합니다.

7. 필요한 경우 조직의 이름 또는 ID를 입력합니다.

Red Hat 고객 포털에서 두 개 이상의 조직에 속해 있는 경우 조직 이름 또는 조직 ID를 추가해야 합니다. org ID를 가져오려면 Red Hat 연락처로 이동하십시오.

- 시스템을 Red Hat Insights에 연결하지 않으려면 Insights 확인란을 선택 취소합니다.

8. Register 버튼을 클릭합니다.

이제 Red Hat Enterprise Linux 시스템이 성공적으로 등록되었습니다.

4.3. GNOME에 RED HAT 계정을 사용하여 시스템 등록

다음 절차에 따라 Red Hat 계정에 시스템을 등록합니다.

사전 요구 사항

- Red Hat 고객 포털에서 유효한 계정.

  새 사용자 등록은 Create a Red Hat 로그인 페이지에서 참조하십시오.

절차

1. 오른쪽 상단 화면 모서리에서 액세스할 수 있는 시스템 메뉴로 이동하여 Settings (설정) 아이콘을 클릭합니다.

2. 세부 정보 섹션에서 등록을 클릭합니다.

3. Register Server 를 선택합니다.

4. Red Hat 서버를 사용하지 않는 경우 URL 필드에 서버 주소를 입력합니다.

5. 등록 유형 메뉴에서 Red Hat 계정을 선택합니다.

6. 등록 세부 정보:

- 로그인 필드에 **Red Hat** 계정 사용자 이름을 입력합니다.
- 암호 필드에 **Red Hat** 계정 암호를 입력합니다.
- 조직 필드에 조직 이름을 입력합니다.

7. 등록을 클릭합니다.

4.4. GNOME에서 활성화 키를 사용하여 시스템 등록

시스템을 활성화 키로 등록하려면 다음 절차의 단계를 수행합니다. 조직 관리자에서 활성화 키를 가져올 수 있습니다.

사전 요구 사항

- 활성화 키 또는 키.

  새 활성화 키를 생성하려면 활성화 키 페이지를 참조하십시오.

절차

1. 오른쪽 상단 화면 모서리에서 액세스할 수 있는 시스템 메뉴로 이동하여 **Settings** (설정) 아이콘을 클릭합니다.
2. 세부 정보 섹션에서 등록을 클릭합니다.
3. **Register Server** 를 선택합니다.
4. **Red Hat** 서버를 사용하지 않는 경우 사용자 지정된 서버에 대한 **URL** 을 입력합니다.

5. **등록 유형 메뉴에서 활성화 키를 선택합니다.**

6. **등록 세부 정보:**

   - **활성화 키를 입력합니다.**

     여러 개의 키를 쉼표(,)로 구분합니다.

   - **조직 필드에 조직의 이름 또는 ID를 입력합니다.**

7. **등록을클릭합니다.**

5장. 부팅 시 SYSTEMD 서비스 시작

systemd는 **systemd** 유닛의 개념을 소개하는 **Linux** 운영 체제용 시스템 및 서비스 관리자입니다.

이 섹션에서는 부팅 시 서비스가 활성화되거나 비활성화되었는지 확인하는 방법에 대한 정보를 제공합니다. 또한 웹 콘솔을 통해 서비스를 관리하는 방법에 대해서도 설명합니다.

5.1. 서비스 활성화 또는 비활성화

설치 프로세스 중 이미 부팅 시 활성화되거나 비활성화되었는지 확인할 수 있습니다. 설치된 운영 체제에서 서비스를 활성화하거나 비활성화할 수도 있습니다.

이 섹션에서는 이미 설치된 운영 체제에서 해당 서비스를 활성화 또는 비활성화하는 단계에 대해 설명합니다.

사전 요구 사항

- 시스템에 대한 루트 액세스 권한이 있어야 합니다.

절차

1. 서비스를 활성화하려면 **enable** 옵션을 사용합니다.

```
# systemctl enable service_name
```

service_name 을 활성화할 서비스로 교체합니다.

단일 명령에서 서비스를 활성화하고 시작할 수도 있습니다.

```
# systemctl enable --now service_name
```

2. 서비스를 비활성화하려면 **disable** 옵션을 사용합니다.

```
# systemctl disable service_name
```

service_name 을 비활성화할 서비스로 교체합니다.

주의

이전에 마스킹한 서비스를 활성화할 수 없습니다. 먼저 마스킹을 해제해야 합니다.

```
# systemctl unmask service_name
```

5.2. RHEL 웹 콘솔에서 서비스 관리

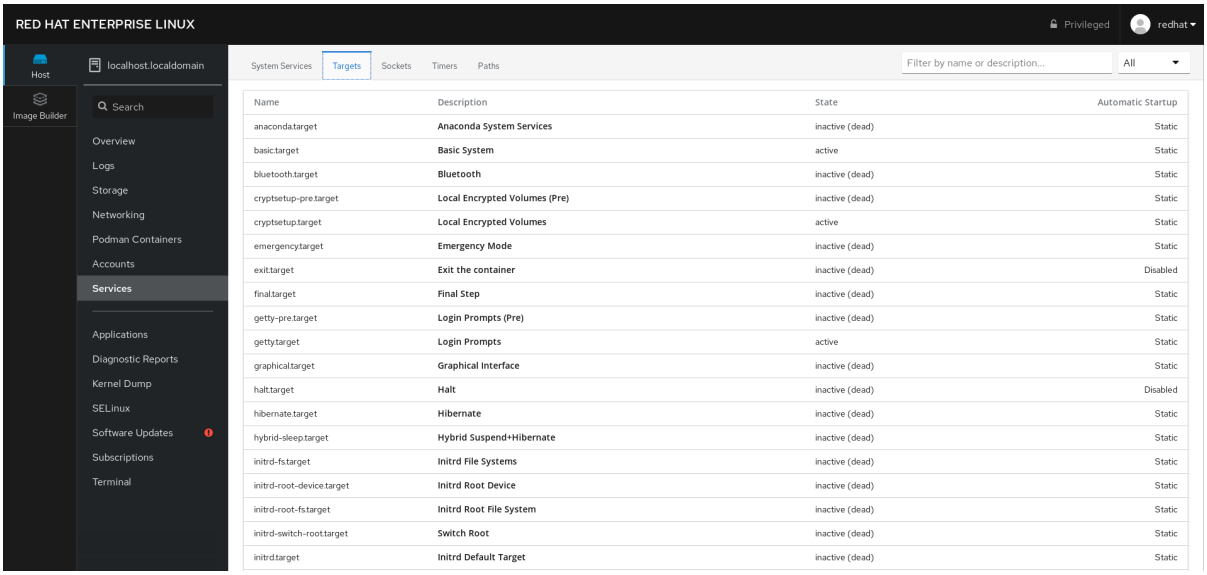
이 섹션에서는 웹 콘솔을 사용하여 서비스를 활성화 또는 비활성화하는 방법을 설명합니다. **systemd** 대상, 서비스, 소켓, 타이머 및 경로를 관리할 수 있습니다. 서비스 상태, 시작 또는 중지, 활성화 또는 비활성화도 확인할 수 있습니다.

사전 요구 사항

- 시스템에 대한 루트 액세스 권한이 있어야 합니다.

절차

1. 기본 설정의 웹 브라우저에서 **localhost:9090** 을 엽니다.
2. 시스템에서 루트 인증 정보를 사용하여 웹 콘솔에 로그인합니다.
3. 웹 콘솔 패널을 표시하려면 창의 왼쪽 상단에 있는 호스트 아이콘을 클릭합니다.



4. **메뉴에서 서비스를 클릭합니다.**

   systemd 대상, 서비스, 소켓, 타이머 및 경로를 관리할 수 있습니다.

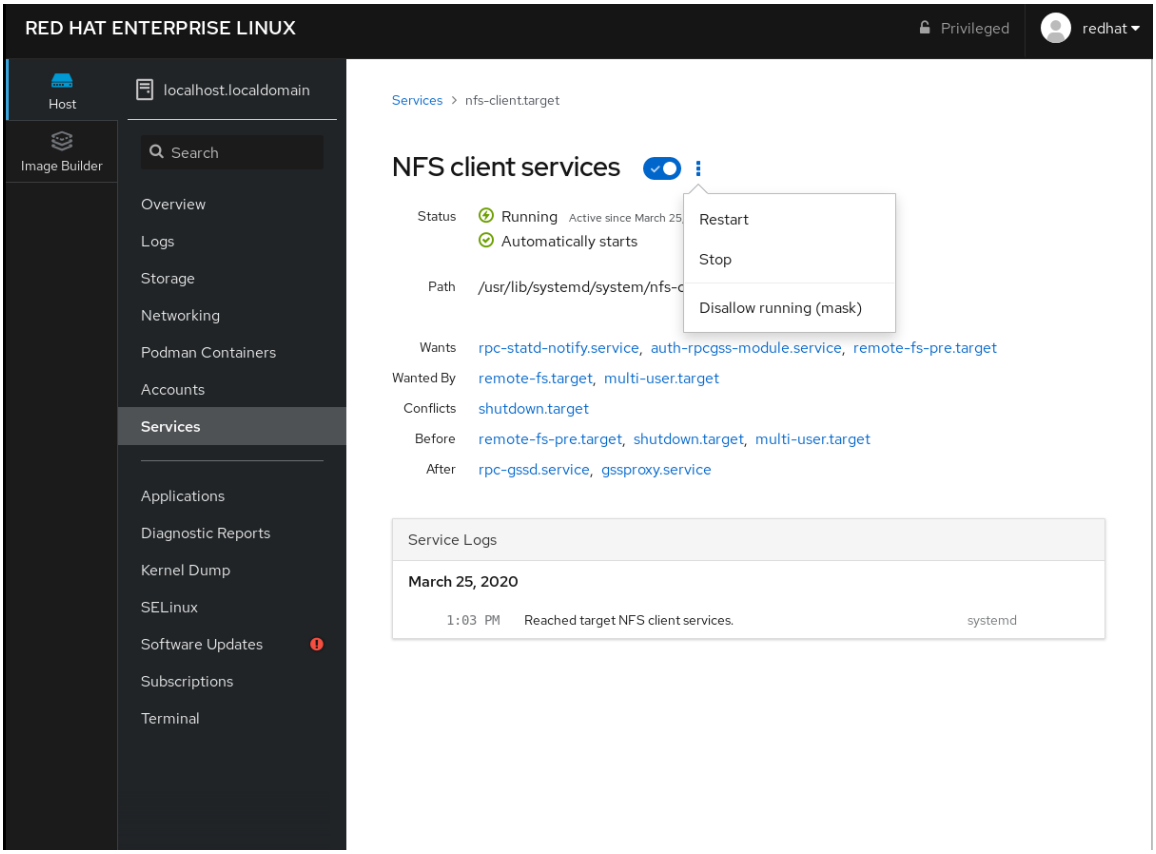
5. **예를 들어 서비스 NFS 클라이언트 서비스를 관리하려면 다음을 수행합니다.**

   a. **대상을 클릭합니다.**

   b. **NFS 클라이언트 서비스를 선택합니다.**

   c. **서비스를 활성화하거나 비활성화하려면 Toogle 버튼을 클릭합니다.**

   d. **서비스를 중지하려면 octets 버튼을 클릭하고 Stop (중지) 옵션을 선택합니다.**


RED HAT ENTERPRISE LINUX
Privileged
redhat
Host
Image Builder
localhost.localdomain
Search
Overview
Logs
Storage
Networking
Podman Containers
Accounts
Services
Applications
Diagnostic Reports
Kernel Dump
SELinux
Software Updates
Subscriptions
Terminal
Services > nfs-client.target
NFS client services
Status
Running
Active since March 25,
Automatically starts
Path
/usr/lib/systemd/system/nfs-c
Restart
Stop
Disallow running (mask)
Wants
rpc-statd-notify.service, auth-rpcgss-module.service, remote-fs-pre.target
Wanted By
remote-fs.target, multi-user.target
Conflicts
shutdown.target
Before
remote-fs-pre.target, shutdown.target, multi-user.target
After
rpc-gssd.service, gssproxy.service
Service Logs
March 25, 2020
1:03 PM
Reached target NFS client services.
systemd

6장. 시스템 보안 구성

컴퓨터 보안은 컴퓨터 시스템과 하드웨어, 소프트웨어, 정보, 서비스를 도용, 손상, 중단 및 잘못으로부터 보호하는 것입니다. 특히 중요한 데이터를 처리하고 비즈니스 트랜잭션을 처리하는 기업이 컴퓨터 보안을 보장하는 것이 중요합니다.

이 섹션에서는 운영 체제 설치 후 구성할 수 있는 기본 보안 기능만 다룹니다.

6.1. FIREWALLD 서비스 활성화

방화벽은 구성된 보안 규칙에 따라 들어오고 나가는 네트워크 트래픽을 모니터링하고 제어하는 네트워크 보안 시스템입니다. 방화벽은 일반적으로 신뢰할 수 있는 보안 내부 네트워크와 다른 외부 네트워크 간의 장벽을 설정합니다.

Red Hat Enterprise Linux에서 방화벽을 제공하는 **firewalld** 서비스는 설치 중에 자동으로 활성화됩니다.

firewalld 서비스를 활성화하려면 다음 절차를 따르십시오.

절차

- **firewalld** 의 현재 상태 표시:

```
$ systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset:
enabled)
   Active: inactive (dead)
...
```

- **firewalld** 가 활성화되어 실행되지 않으면 **root** 사용자로 전환하고 **firewalld** 서비스를 시작하고 시스템을 다시 시작한 후 자동으로 시작합니다.

```
# systemctl enable --now firewalld
```

검증 단계

- **firewalld** 가 실행 중이고 활성화되어 있는지 확인합니다.

```
$ systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset:
enabled)
   Active: active (running)
...
```

추가 리소스

- firewalld 사용 및 구성
- man firewalld(1)

6.2. 기본 SELINUX 설정 관리

SELinux(Security-hardened Linux)는 어떤 프로세스가 어떤 파일, 디렉터리 및 포트에 액세스할 수 있는지 결정하는 추가 시스템 보안 계층입니다. 이러한 권한은 SELinux 정책에 정의되어 있습니다. 정책은 SELinux 보안 엔진을 안내하는 일련의 규칙입니다.

SELinux에는 다음 두 가지 상태가 있습니다.

- disabled
- enabled

SELinux가 활성화되면 다음 모드 중 하나에서 실행됩니다.

- enabled
  - enforcing
  - 허용

강제 모드에서 SELinux는 로드된 정책을 적용합니다. SELinux는 SELinux 정책 규칙에 따라 액세스

를 거부하고 명시적으로 허용되는 상호 작용만 활성화합니다. 강제 모드는 가장 안전한 **SELinux** 모드이며 설치 후 기본 모드입니다.

허용 모드에서 **SELinux**는 로드된 정책을 적용하지 않습니다. **SELinux**는 액세스를 거부하지 않지만 규칙을 중단하는 작업을 **/var/log/audit/audit.log** 로그에 보고합니다. 허용 모드는 설치 중에 기본 모드입니다. **Permissive** 모드는 예를 들어 문제 해결과 같은 특정 경우에 유용합니다.

추가 리소스

- **SELinux** 사용

6.3. SELINUX의 필수 상태 확인

기본적으로 **SELinux**는 강제 모드로 작동합니다. 그러나 특정 시나리오에서는 **SELinux**를 허용 모드로 설정하거나 비활성화할 수도 있습니다.

중요

Red Hat은 시스템을 강제 모드로 유지할 것을 권장합니다. 디버깅 목적으로 **SELinux**를 허용 모드로 설정할 수 있습니다.

시스템에서 **SELinux**의 상태 및 모드를 변경하려면 다음 절차를 따르십시오.

절차

1. 현재 **SELinux** 모드를 표시합니다.

```
$ getenforce
```

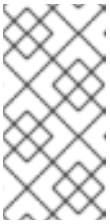
2. **SELinux**를 일시적으로 설정하려면 다음을 수행합니다.

   a. 강제 모드로 전환하려면 다음을 수행합니다.

```
# setenforce Enforcing
```

b. 허용 모드의 경우 다음을 수행합니다.

```
# setenforce Permissive
```



참고

재부팅 후 **SELinux** 모드는 **/etc/selinux/config** 구성 파일에 지정된 값으로 설정됩니다.

3. 재부팅 시 지속되도록 **SELinux** 모드를 설정하려면 **/etc/selinux/config** 구성 파일에서 **SELINUX** 변수를 수정합니다.

예를 들어 **SELinux**를 강제 모드로 전환하려면 다음을 수행합니다.

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=enforcing
...
```

주의

SELinux를 비활성화하면 시스템 보안이 줄어듭니다. 메모리 누수 및 경쟁 조건으로 인해 커널 패닉이 발생할 수 있으므로 **/etc/selinux/config** 파일에서 **SELINUX=disabled** 옵션을 사용하여 **SELinux**를 비활성화하지 마십시오. 대신 커널 명령줄에 **selinux=0** 매개 변수를 추가하여 **SELinux**를 비활성화합니다. 자세한 내용은 부팅 시 **SELinux** 모드 변경을 참조하십시오.

추가 리소스

- **SELinux** 상태 및 모드 변경

6.4. 추가 리소스

- SSH 키 쌍 생성
- 키 기반 인증을 위한 OpenSSH 서버 설정
- 보안 강화
- SELinux 사용
- 네트워크 보안

7장. 사용자 계정 관리 시작하기

Red Hat Enterprise Linux는 다중 사용자 운영 체제로, 다른 컴퓨터에서 여러 사용자가 한 시스템에 설치된 단일 시스템에 액세스할 수 있습니다. 모든 사용자는 자체 계정으로 운영되며 사용자 계정을 관리하여 **Red Hat Enterprise Linux** 시스템 관리의 핵심 요소를 나타냅니다.

다음은 다양한 유형의 사용자 계정입니다.

- 일반 사용자 계정:

  특정 시스템의 사용자를 위해 일반 계정이 생성됩니다. 이러한 계정은 일반 시스템 관리 중에 추가, 제거 및 수정할 수 있습니다.

- 시스템 사용자 계정:

  시스템 사용자 계정은 시스템의 특정 애플리케이션 식별자를 나타냅니다. 이러한 계정은 일반적으로 소프트웨어 설치 시에만 추가되거나 조작되며 나중에 수정되지 않습니다.

  주의

  시스템에서 로컬로 시스템 계정을 사용할 수 있다고 가정합니다. **LDAP** 구성 인스턴스에서와 같이 이러한 계정이 원격으로 구성되고 제공되는 경우 시스템 중단 및 서비스 시작 오류가 발생할 수 있습니다.

  시스템 계정의 경우 **1000** 미만의 사용자 **ID**가 예약됩니다. 일반 계정의 경우 **1000**부터 시작하는 **ID**를 사용할 수 있습니다. 그러나 권장 사례는 **5000**에서 시작하는 **ID**를 할당하는 것입니다. **ID** 할당은 **/etc/login.defs** 파일을 참조하십시오.

- **group:**

  그룹은 특정 파일에 대한 액세스 권한 부여와 같은 공통 목적을 위해 여러 사용자 계정을 함께 연결하는 엔티티입니다.

7.1. 명령줄 도구를 사용하여 계정 및 그룹 관리

이 섹션에서는 사용자 계정 및 그룹을 관리하는 기본 명령줄 도구에 대해 설명합니다.

- 사용자 및 그룹 **ID**를 표시하려면 다음을 수행합니다.

```
$ id
uid=1000(example.user) gid=1000(example.user) groups=1000(example.user),10(wheel)
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

- 새 사용자 계정을 생성하려면 다음을 수행합니다.

```
# useradd example.user
```

- ***example.user*** 에 속하는 사용자 계정에 새 암호를 할당하려면 다음을 수행합니다.

```
# passwd example.user
```

- 그룹에 사용자를 추가하려면 다음을 수행합니다.

```
# usermod -a -G example.group example.user
```

추가 리소스

- **man useradd(8), man passwd(1), man usermod(8)**

7.2. 웹 콘솔에서 관리되는 시스템 사용자 계정

RHEL 웹 콘솔에 사용자 계정이 표시되는 경우 다음을 수행할 수 있습니다.

- 시스템에 액세스할 때 사용자를 인증합니다.
- 시스템에 대한 액세스 권한을 설정합니다.

RHEL 웹 콘솔은 시스템에 있는 모든 사용자 계정을 표시합니다. 따라서 웹 콘솔에 처음 로그인한 직후 적어도 하나의 사용자 계정을 볼 수 있습니다.

RHEL 웹 콘솔에 로그인한 후 다음 작업을 수행할 수 있습니다.

- 새 사용자 계정을 생성합니다.
- 매개 변수를 변경합니다.
- 계정을 잠급니다.
- 사용자 세션을 종료합니다.

7.3. 웹 콘솔을 사용하여 새 계정 추가

다음 단계를 사용하여 시스템에 사용자 계정을 추가하고 RHEL 웹 콘솔을 통해 계정에 관리 권한을 설정합니다.

사전 요구 사항

- RHEL 웹 콘솔을 설치하고 액세스할 수 있어야 합니다. 자세한 내용은 웹 콘솔 설치를 참조하십시오.

절차

1. RHEL 웹 콘솔에 로그인합니다.
2. 계정을 클릭합니다.
3. 새 계정 만들기를 클릭합니다.

1. 전체 이름 필드에 사용자의 전체 이름을 입력합니다.

RHEL 웹 콘솔은 전체 이름에서 사용자 이름을 자동으로 권장하고 사용자 이름 필드에 입력합니다. 첫 번째 이름의 첫 글자와 전체 성으로 구성된 원래 이름 지정 규칙을 사용하지 않으려면 제안을 업데이트합니다.

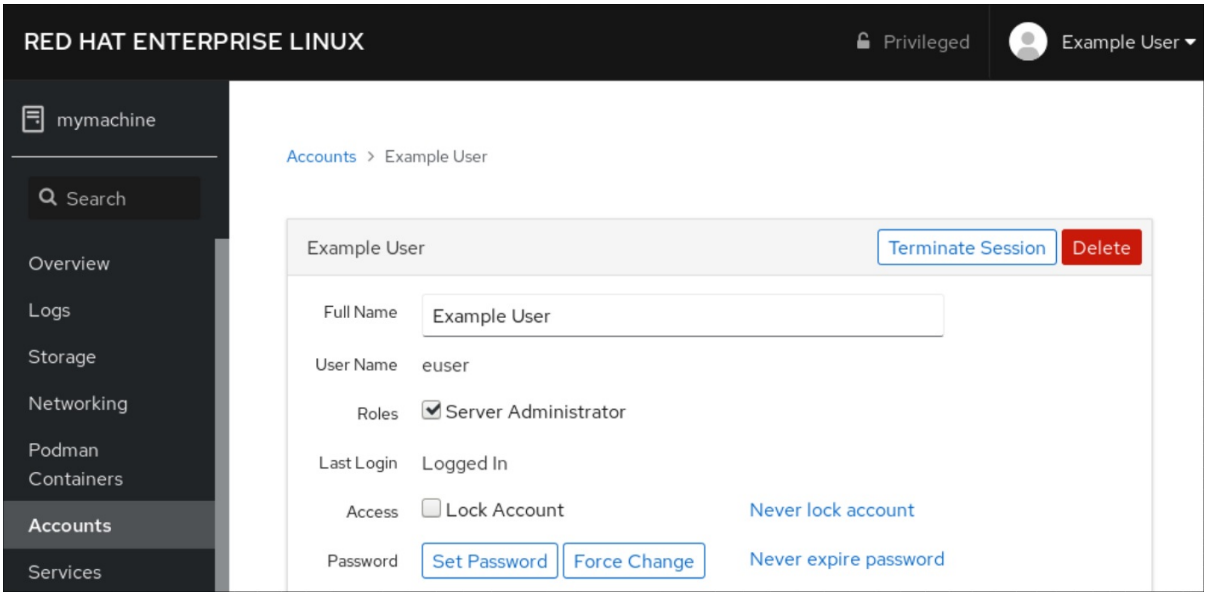
2. Password/Confirm 필드에 암호를 입력하고 암호가 올바른지 확인하도록 다시 입력합니다.

필드 아래에 배치된 색상 표시줄에는 암호가 약한 사용자를 생성할 수 없는 입력한 암호의 보안 수준이 표시됩니다.

1. 만들기를 클릭하여 설정을 저장하고 대화 상자를 종료합니다.

2. 새로 생성된 계정을 선택합니다.

3. Roles (역할) 항목에서 Server Administrator 를 선택합니다.



이제 계정 설정에서 새 계정을 볼 수 있으며 자격 증명을 사용하여 시스템에 연결할 수 있습니다.

8장. 이후 분석을 위해 크래시된 커널 덤프

시스템이 충돌하는 이유를 분석하려면 **kdump** 서비스를 사용하여 나중에 분석할 수 있도록 시스템 메모리 콘텐츠를 저장할 수 있습니다. 이 섹션에서는 **kdump** 에 대한 간략한 소개와 **RHEL** 웹 콘솔을 사용하여 **kdump** 구성 또는 해당 **RHEL** 시스템 역할을 사용하는 방법에 대한 정보를 제공합니다.

8.1. KDUMP란?

kdump 는 크래시 덤프 메커니즘을 제공하는 서비스입니다. 이 서비스를 사용하면 분석을 위해 시스템 메모리의 내용을 저장할 수 있습니다. **kdump** 는 **kexec** 시스템 호출을 사용하여 재부팅하지 않고 두 번째 *커널*(커널 캡처 커널)로 부팅한 다음 *충돌* 된 커널의 메모리 콘텐츠를 캡처하여 파일에 저장합니다. *두 번째 커널은 시스템 메모리의 예약된 부분에 있습니다.*

중요

*커널 크래시 덤프는 (중요 버그) 시스템 오류 발생시 사용 가능한 유일한 정보가 될 수 있습니다. 따라서 운영 **kdump** 는 미션 크리티컬한 환경에서 중요합니다. **Red Hat**은 시스템 관리자가 일반 커널 업데이트 주기에서 **kexec-tools** 를 정기적으로 업데이트하고 테스트하는 것을 권장합니다. 이는 새로운 커널 기능이 구현되는 경우 특히 중요합니다.*

*머신의 설치된 모든 커널 또는 지정된 커널에서만 **kdump** 를 활성화할 수 있습니다. 이 기능은 시스템에 사용된 커널이 여러 개 있을 때 유용하며 그 중 일부는 충돌할 수 있다는 우려가 없을 정도로 안정적입니다.*

***kdump** 가 설치되면 기본 **/etc/kdump.conf** 파일이 생성됩니다. 파일에는 기본 최소 **kdump** 구성이 포함되어 있습니다. 이 파일을 편집하여 **kdump** 구성을 사용자 지정할 수 있지만 필수는 아닙니다.*

8.2. 웹 콘솔에서 KDUMP 메모리 사용량 및 대상 위치 설정

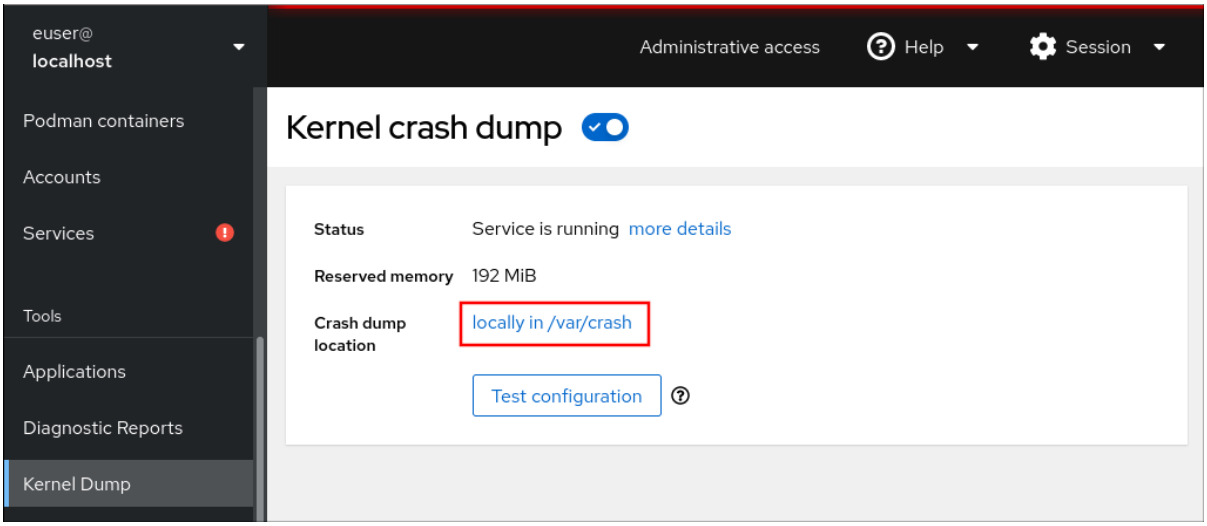
*아래 절차는 **RHEL** 웹 콘솔 인터페이스에서 **Kernel Dump(커널 덤프)** 탭을 사용하여 **kdump** 커널에 대해 예약된 메모리 양을 구성하는 방법을 보여줍니다. 이 절차에서는 **vmcore** 덤프 파일의 대상 위치 및 구성을 테스트하는 방법도 설명합니다.*

절차

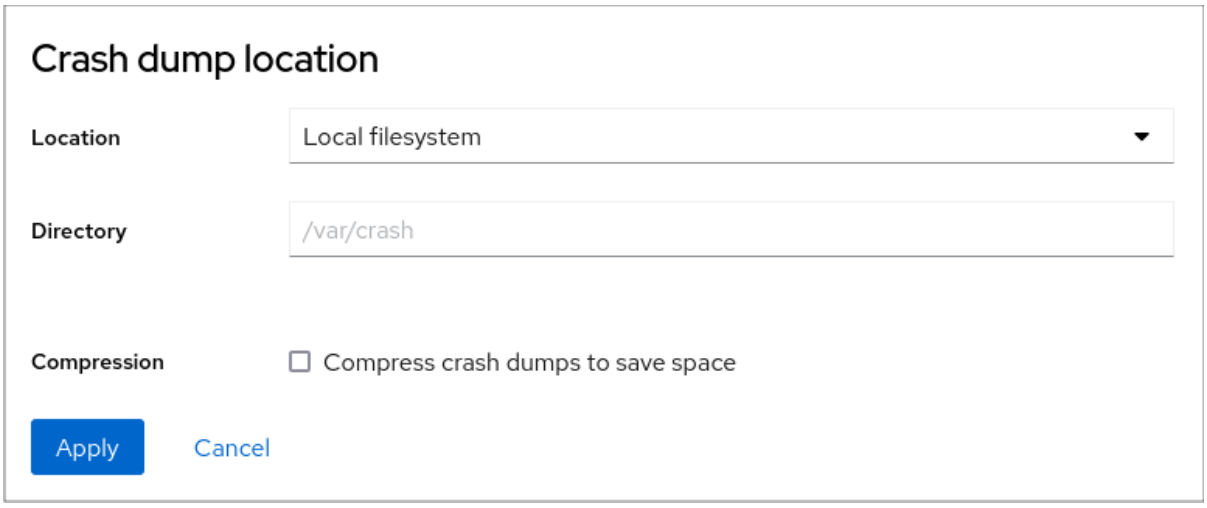
1. *커널 덤프 탭을 열고 **kdump** 서비스를 시작합니다.*

2. *명령줄을 사용하여 **kdump** 메모리 사용량을 구성합니다.*

3. ***Crash** 덤프 위치 옵션 옆에 있는 링크를 클릭합니다.*



4. *드롭다운 메뉴에서 로컬 파일 시스템 옵션을 선택하고 덤프를 저장할 디렉터리를 지정합니다.*



- *또는 드롭다운에서 **Remote over SSH** 옵션을 선택하여 **SSH** 프로토콜을 사용하여 **vmcore**를 원격 시스템으로 전송합니다.*

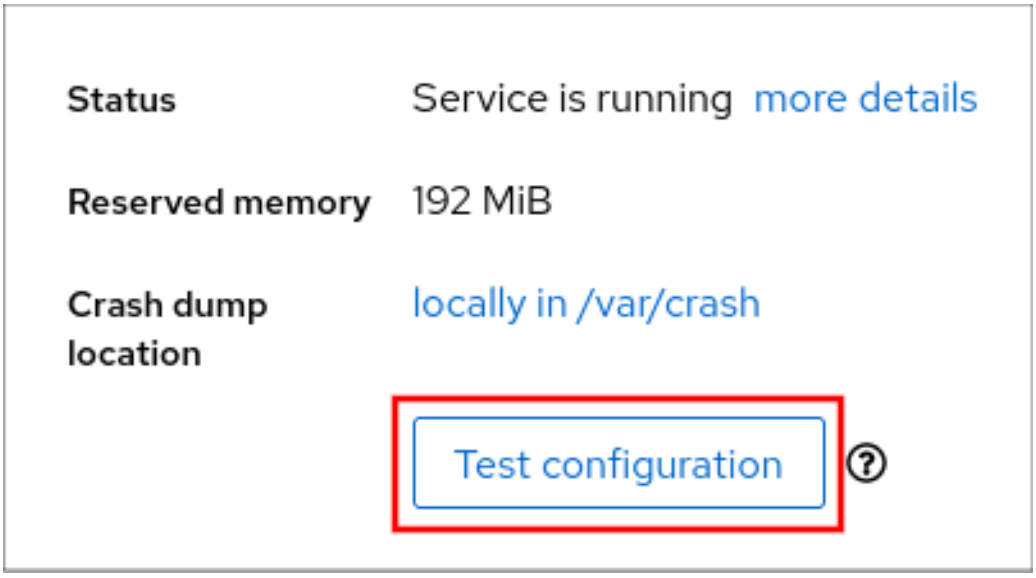
  *서버,**ssh** 키 및 디렉터리 필드 를 원격 머신 주소, **ssh** 키 위치 및 대상 디렉터리로 채웁니다.*

- *또 다른 선택 사항은 드롭다운에서 **Remote over NFS** 옵션을 선택하고 **Mount** 필드를 채워 **NFS** 프로토콜을 사용하여 **vmcore**를 원격 시스템으로 보내는 것입니다.*

참고

*압축 확인란을 선택하여 **vmcore** 파일의 크기를 줄입니다.*

5. *커널을 충돌하여 구성을 테스트합니다.*



a. *테스트 구성 을 클릭합니다.*

b. ***Test kdump** 설정 필드에서 **Crash** 시스템을 클릭합니다.*

주의

이 단계는 커널 실행을 중단시키고 시스템 충돌 및 데이터 손실을 초래합니다.

추가 리소스

- *지원되는 **kdump** 대상*
- ***OpenSSH**로 두 시스템 간의 보안 통신 사용*

8.3. RHEL 시스템 역할을 사용하는 KDUMP

RHEL 시스템 역할은 여러 RHEL 시스템을 원격으로 관리할 수 있는 일관된 구성 인터페이스를 제공하는 Ansible 역할 및 모듈 컬렉션입니다. kdump 역할을 사용하면 여러 시스템에서 기본 커널 덤프 매개변수를 설정할 수 있습니다.

주의

kdump 역할은 /etc/kdump.conf 파일을 교체하여 관리 호스트의 kdump 설정을 완전히 대체합니다. 또한 kdump 역할이 적용되면 /etc/sysconfig/kdump 파일을 교체하여 역할 변수에서 지정하지 않은 경우에도 이전 kdump 설정도 교체됩니다.

다음 예제 Playbook은 kdump 시스템 역할을 적용하여 크래시 덤프 파일의 위치를 설정하는 방법을 보여줍니다.

```
---
- hosts: kdump-test
  vars:
    kdump_path: /var/crash
  roles:
    - rhel-system-roles.kdump
```

kdump 역할 변수에 대한 자세한 참조를 보려면 rhel-system-roles 패키지를 설치하고 /usr/share/doc/rhel-system-roles/kdump 디렉터리의 README.md 또는 README.html 파일을 참조하십시오.

추가 리소스

- RHEL 시스템 역할 소개

8.4. 추가 리소스

- kdump 설치
- 명령줄에서 kdump 설정

- *웹 콘솔에서 kdump 구성*

9장. 시스템 복구 및 복원

기존 백업을 사용하여 시스템을 복구 및 복구하기 위해 **Red Hat Enterprise Linux**는 **Relax-and-Recover(ReaR)** 유틸리티를 제공합니다.

유틸리티를 재해 복구 솔루션으로 사용하고 시스템 마이그레이션에도 사용할 수 있습니다.

유틸리티를 사용하면 다음 작업을 수행할 수 있습니다.

- 이미지를 사용하여 부팅 가능한 이미지를 생성하고 기존 백업에서 시스템을 복원합니다.
- 원래 스토리지 레이아웃을 복제합니다.
- 사용자 및 시스템 파일을 복원합니다.
- 시스템을 다른 하드웨어로 복원합니다.

또한 재해 복구를 위해 특정 백업 소프트웨어를 **ReaR**과 통합할 수도 있습니다.

ReaR 설정에는 다음과 같은 상위 수준 단계가 포함됩니다.

1. **ReaR**을 설치합니다.
2. **ReaR** 구성 파일을 수정하여 백업 방법 세부 정보를 추가합니다.
3. 복구 시스템을 생성합니다.
4. 백업 파일을 생성합니다.

9.1. REAR 설정

다음 단계에 따라 **Relax-and-Recover(ReaR)** 유틸리티를 사용하는 패키지를 설치하고 복구 시스템을 생성하고 백업을 구성 및 생성합니다.

사전 요구 사항

- 백업 복원 계획에 따라 필요한 구성이 준비되었습니다.

  ReaR과 함께 완전 통합 및 기본 제공 방법인 **NETFS** 백업 메서드를 사용할 수 있습니다.

절차

1. 다음 명령을 실행하여 **ReaR** 유틸리티를 설치합니다.

```
# dnf install rear
```

2. 선택한 편집기에서 **ReaR** 구성 파일을 수정합니다. 예를 들면 다음과 같습니다.

```
# vi /etc/rear/local.conf
```

3. **/etc/rear/local.conf** 에 백업 설정 세부 정보를 추가합니다. 예를 들어 **NETFS** 백업 방법의 경우 다음 행을 추가합니다.

```
BACKUP=NETFS
BACKUP_URL=backup.location
```

   backup.location 을 백업 위치의 **URL**로 바꿉니다.

4. 새 파일이 생성될 때 이전 백업 아카이브를 유지하도록 **ReaR**을 구성하려면 구성 파일에 다음 행도 추가합니다.

```
NETFS_KEEP_OLD_BACKUP_COPY=y
```

5. 백업을 증분 방식으로 만들려면 변경된 파일만 각 실행 시 백업됩니다.

```
BACKUP_TYPE=incremental
```

6. 복구 시스템을 생성합니다.

```
# rear mkrescue
```

7. 복원 계획에 따라 백업을 수행합니다. 예를 들어 NETFS 백업 방법의 경우 다음 명령을 실행합니다.

```
# rear mkbackuponly
```

또는 다음 명령을 실행하여 단일 단계에서 복구 시스템 및 백업을 생성할 수 있습니다.

```
# rear mkbackup
```

이 명령은 **rear mkrescue** 및 **rear mk backuponly** 명령의 기능을 결합합니다.

9.2. 64비트 IBM Z 아키텍처에서 REAR RESCUE 이미지 사용

Basic Relax and Recover (ReaR) 기능은 이제 64비트 **IBM Z** 아키텍처에서 기술 프리뷰로 제공됩니다. **IBM Z**에서 **z/VM** 환경에서만 복구 이미지를 생성할 수 있습니다. **LPAR(Logical partitions)** 백업 및 복구는 테스트되지 않았습니다.

중요

64비트 **IBM Z** 아키텍처는 기술 프리뷰 기능 전용입니다. 기술 프리뷰 기능은 **Red Hat** 프로덕션 서비스 수준 계약(**SLA**)에서 지원되지 않으며 기능적으로 완전하지 않을 수 있습니다. 따라서 프로덕션 환경에서 사용하는 것은 권장하지 않습니다. 이러한 기능을 사용하면 향후 제품 기능을 조기에 이용할 수 있어 개발 과정에서 고객이 기능을 테스트하고 피드백을 제공할 수 있습니다. **Red Hat** 기술 프리뷰 기능의 지원 범위에 대한 자세한 내용은 https://access.redhat.com/support/offerings/techpreview 를 참조하십시오.

현재 사용 가능한 출력 방법은 초기 프로그램 로드(**IPL**)입니다. **IPL**은 **zIPL** 부트로더와 함께 사용할 수 있는 커널 및 초기 램디스크(**initrd**)를 생성합니다.

사전 요구 사항

- **Rear**가 설치되어 있습니다.

- **ReaR**을 설치하려면 **dnf install rear** 명령을 실행합니다.

절차

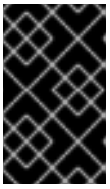
다음 변수를 **/etc/rear/local.conf** 에 추가하여 **64**비트 **IBM Z** 아키텍처에서 복구 이미지를 생성하도록 **ReaR**을 구성합니다.

1. **IPL** 출력 방법을 구성하려면 **OUTPUT=IPL** 을 추가합니다.

2. 백업 방법 및 대상을 구성하려면 **RuntimeClass** 및 **octets _URL** 변수를 추가합니다. 예를 들면 다음과 같습니다.

```
BACKUP=NETFS

BACKUP_URL=nfs://<nfsserver name>/<share path>
```



중요

로컬 백업 스토리지는 현재 **64**비트 **IBM Z** 아키텍처에서 지원되지 않습니다.

3. 선택적으로 커널 및 **initrd** 파일을 저장할 수 있도록 **OUTPUT_URL** 변수를 구성할 수도 있습니다. 기본적으로 **OUTPUT_URL** 은 **etcdctl_URL**에 맞게 조정됩니다.

4. 백업 및 복구 이미지 생성을 수행하려면 다음을 수행합니다.

```
rear mkbackup
```

5. 이렇게 하면 **EgressIP _URL** 또는 **OUTPUT_URL** (설정된 경우) 변수에서 지정한 위치에 커널 및 **initrd** 파일이 생성되고 지정된 백업 방법을 사용하여 백업을 생성합니다.

6. 시스템을 복구하려면 **3**단계에서 만든 **ReaR** 커널 및 **initrd** 파일을 사용하고, **zipl** 부트 로더, 커널, **initrd**를 사용하여 준비한 **zipl boot loader, kernel** 및 **initrd**를 사용하여 준비한 **FCP(Direct Attached Storage Device)** 또는 **FCP(Fibre Channel Protocol)**에서 부팅하십시오. 자세한 내용은 **Prepared DASD** 사용을 참조하십시오.

7. **rescue** 커널 및 **initrd** 가 부팅되면 **ReaR rescue** 환경을 시작합니다. 시스템 복구를 진행합니

다.

주의

현재 복구 프로세스는 시스템에 연결된 모든 **DASD(Direct Attached Storage** 장치)를 다시 포맷합니다. 시스템 스토리지 장치에 중요한 데이터가 있는 경우 시스템 복구를 시도하지 마십시오. 여기에는 복구 환경으로 부팅하는 데 사용된 **zipl** 부트로더, **ReaR** 커널 및 **initrd**가 준비된 장치도 포함됩니다. 복사본을 유지해야 합니다.

추가 리소스

- z/VM에 설치
- 준비 DASD 사용

10장. 로그 파일을 사용하여 문제 해결

로그 파일에는 커널, 서비스 및 커널에서 실행되는 애플리케이션을 포함하여 시스템에 대한 메시지가 포함되어 있습니다. 여기에는 문제를 해결하거나 시스템 기능을 모니터링하는 데 도움이 되는 정보가 포함되어 있습니다. Red Hat Enterprise Linux의 로깅 시스템은 내장 syslog 프로토콜을 기반으로 합니다. 특정 프로그램은 이 시스템을 사용하여 이벤트를 기록하고 이를 로그 파일로 구성하며 운영 체제를 감사하고 다양한 문제를 해결할 때 유용합니다.

10.1. SYSLOG 메시지를 처리하는 서비스

다음 두 서비스는 syslog 메시지를 처리합니다.

- systemd-journald 데몬
- Rsyslog 서비스

systemd-journald 데몬은 다양한 소스에서 메시지를 수집하고 추가 처리를 위해 Rsyslog 에 전달합니다. systemd-journald 데몬은 다음 소스에서 메시지를 수집합니다.

- 커널
- 부팅 과정의 초기 단계
- 데몬의 표준 및 오류 출력 시작 및 실행
- syslog

Rsyslog 서비스는 syslog 메시지를 유형 및 우선 순위에 따라 정렬하고 /var/log 디렉터리의 파일에 씁니다. /var/log 디렉터리는 로그 메시지를 영구적으로 저장합니다.

10.2. SYSLOG 메시지를 저장하는 하위 디렉터리

/var/log 디렉터리 아래의 다음 하위 디렉터리에 syslog 메시지를 저장합니다.

- */var/log/message* - 다음을 제외한 모든 **syslog** 메시지
- */var/log/secure* - 보안 및 인증 관련 메시지 및 오류
- */var/log/maillog* - 메일 서버 관련 메시지 및 오류
- */var/log/cron* - 주기적으로 실행한 작업과 관련된 로그 파일
- */var/log/boot.log* - 시스템 시작과 관련된 로그 파일

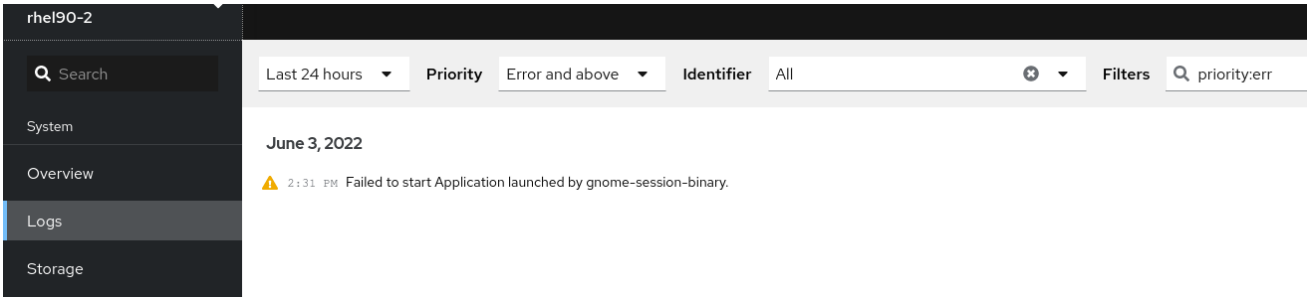
10.3. 웹 콘솔을 사용하여 로그 파일 검사

이 절차의 단계에 따라 **RHEL** 웹 콘솔을 사용하여 로그 파일을 검사합니다.

절차

1. **RHEL** 웹 콘솔에 로그인합니다. 자세한 내용은 *웹 콘솔로의 로깅* 을 참조하십시오.
2. 로그를 클릭합니다.

그림 10.1. RHEL 9 웹 콘솔에서 로그 파일 검사



10.4. 명령줄을 사용하여 로그 보기

journal는 로그 파일을 보고 관리하는 데 도움이 되는 **systemd**의 구성 요소입니다. 다른 시스템과 밀접하게 통합되어 기존 로깅과 연결된 문제를 해결하고 로그 파일에 대한 다양한 로깅 기술 및 액세스 관리를 지원합니다.

journalctl 명령을 사용하여 명령줄을 사용하여 시스템 저널의 메시지를 볼 수 있습니다. 예를 들면 다음과 같습니다.

```
$ journalctl -b | grep kvm
May 15 11:31:41 localhost.localdomain kernel: kvm-clock: Using msrs 4b564d01 and 4b564d00
May 15 11:31:41 localhost.localdomain kernel: kvm-clock: cpu 0, msr 76401001, primary cpu clock
...
```

표 10.1. 시스템 정보 보기

명령	설명
journalctl	수집된 모든 저널 항목을 표시합니다.
journalctl FILEPATH	특정 파일과 관련된 로그를 표시합니다. 예를 들어 **journalctl /dev/sda** 명령은 **/dev/sda** 파일 시스템과 관련된 로그를 표시합니다.
journalctl -b	현재 부팅에 대한 로그를 표시합니다.
journalctl -k -b -1	현재 부팅에 대한 커널 로그를 표시합니다.

표 10.2. 특정 서비스에 대한 정보 보기

명령	설명
journalctl -b _SYSTEMD_UNIT=foo	필터 로그는 "foo" **systemd** 서비스와 일치하는 항목을 확인합니다.
journalctl -b _SYSTEMD_UNIT=foo _PID=number	일치 항목이 결합됩니다. 예를 들어 이 명령은 **foo** 및 PID **번호와** 일치하는 **systemd-units** 의 로그를 표시합니다.
journalctl -b _SYSTEMD_UNIT=foo _PID=number + _SYSTEMD_UNIT=foo1	구분 기호 "+"는 논리 OR에서 두 표현식을 결합합니다. 예를 들어, 이 명령은 **foo** 서비스 프로세스의 모든 메시지를 **PID** 와 **foo1** 서비스의 모든 메시지(모든 프로세스)와 함께 표시합니다.
journalctl -b _SYSTEMD_UNIT=foo _SYSTEMD_UNIT=foo1	이 명령은 동일한 필드를 참조하는 표현식과 일치하는 모든 항목을 표시합니다. 여기에서 이 명령은 systemd-unit **foo** 또는 systemd-unit **foo1** 과 일치하는 로그를 표시합니다.

표 10.3. 특정 부팅과 관련된 로그 보기

명령	설명
journalctl --list-boots	부팅과 관련된 첫 번째 및 마지막 메시지의 테이블 형식 목록, 해당 ID 및 타임스탬프를 표시합니다. 다음 명령에서 ID를 사용하여 세부 정보를 볼 수 있습니다.
journalctl --boot=ID _SYSTEMD_UNIT=foo	지정된 부팅 ID에 대한 정보를 표시합니다.

10.5. 추가 리소스

- ***man journalctl(1)***
- *원격 로깅 솔루션 구성*

11장. RED HAT 지원 액세스

이 섹션에서는 Red Hat 지원 및 sosreport 를 사용하여 문제를 효과적으로 해결하는 방법을 설명합니다.

Red Hat의 지원을 받으려면 서브스크립션과 함께 제공되는 모든 항목에 액세스할 수 있는 Red Hat 고객 포털 을 사용하십시오.

11.1. RED HAT 고객 포털을 통해 RED HAT 지원 받기

다음 섹션에서는 Red Hat 고객 포털 사용 방법에 대해 설명합니다.

사전 요구 사항

- Red Hat 고객 포털에서 유효한 사용자 계정. Red Hat 로그인 생성을 참조하십시오.
- RHEL 시스템의 활성 서브스크립션입니다.

절차

1. Red Hat 지원에 액세스 :
   a. 새 지원 케이스를 엽니다.
   b. Red Hat 전문가와의 실시간 채팅 시작.
   c. 이메일을 보내거나 전화하여 Red Hat 전문가에게 문의하십시오.

11.2. SOSREPORT를 사용하여 문제 해결

sosreport 명령은 Red Hat Enterprise Linux 시스템에서 구성 세부 정보, 시스템 정보 및 진단 정보를 수집합니다.

다음 섹션에서는 **sosreport** 명령을 사용하여 지원 사례에 대한 보고서를 생성하는 방법을 설명합니다.

사전 요구 사항

- **Red Hat** 고객 포털에서 유효한 사용자 계정. **Red Hat** 로그인 생성을 참조하십시오.
- **RHEL** 시스템의 활성 서브스크립션입니다.
- 지원 케이스 번호입니다.

절차

1. **sos** 패키지를 설치합니다.

```
# dnf install sos
```

참고

Red Hat Enterprise Linux의 기본 최소 설치에는 **sosreport** 명령을 제공하는 **sos** 패키지가 포함되어 있지 않습니다.

2. 보고서를 생성합니다.

```
# sosreport
```

3. 지원 케이스에 보고서를 첨부합니다.

Red Hat 지원 케이스에 파일을 첨부하려면 어떻게 해야 합니까? 자세한 내용은 **Red Hat Knowledgebase** 문서.

보고서를 첨부할 때 관련 지원 케이스 수를 입력하라는 메시지가 표시됩니다.

추가 리소스

- *sosreport는 무엇이며 Red Hat Enterprise Linux에서 하나를 만드는 방법은 무엇입니까?*

12장. SYSTEMD 소개

systemd 는 **Linux** 운영 체제용 시스템 및 서비스 관리자입니다. 이는 **SysV init** 스크립트와 역호환되도록 설계되었으며 부팅 시 시스템 서비스의 병렬 시작, 필요에 따라 데몬 활성화 또는 종속성 기반 서비스 제어 논리와 같은 여러 기능을 제공합니다. **Red Hat Enterprise Linux 7**부터 **systemd** 는 기본 **init** 시스템으로 **Upstart**를 대체합니다.

systemd 는 **systemd** 유닛 의 개념을 도입합니다. 이러한 단위는 다음 표에 나열된 디렉터리 중 하나에 있는 단위 구성 파일로 표시됩니다.

표 **12.1. Systemd** 유닛 파일 위치

디렉터리	설명
/usr/lib/systemd/system/	설치된 RPM 패키지와 함께 배포되는 systemd 유닛 파일.
/run/systemd/system/	런타임 시 생성된 systemd 유닛 파일. 이 디렉터리는 설치된 서비스 장치 파일이 있는 디렉터리보다 우선합니다.
/etc/systemd/system/	**systemctl enable** 에 의해 생성된 systemd 유닛 파일과 서비스 확장에 추가된 유닛 파일. 이 디렉터리는 런타임 장치 파일이 있는 디렉터리보다 우선합니다.

이 단위는 다음에 대한 정보를 캡슐화합니다.

- 시스템 서비스
- 수신 소켓
- **init** 시스템과 관련된 기타 오브젝트

systemd 의 기본 구성은 컴파일 중에 정의되며 **/etc/systemd/system.conf** 의 **systemd** 구성 파일에서 확인할 수 있습니다. 이러한 기본값에서 벗어나지 않도록 하려면 이 파일을 사용하여 **systemd** 단위에 대해 선택된 기본값을 전역적으로 덮어씁니다.

예를 들어 **90**초로 설정된 제한 시간 제한의 기본값을 재정의하려면 **DefaultTimeoutStartSec** 매개변수를 사용하여 필요한 값을 초 단위로 입력합니다.

> ***DefaultTimeoutStartSec=pass:_required value_***

12.1. SYSTEMD 장치 유형

*사용 가능한 **systemd** 장치 유형의 전체 목록은 다음 표를 참조하십시오.*

*표 12.2. 사용 가능한 **systemd** 장치 유형*

단위 유형	파일 확장자	설명
서비스 단위	**.service**	시스템 서비스.
대상 단위	**.target**	systemd 장치 그룹입니다.
자동 마운트 단위	**.automount**	파일 시스템 자동 마운트 지점.
장치 단위	**.device**	커널에서 인식한 장치 파일입니다.
마운트 단위	**.Mount**	파일 시스템 마운트 지점.
경로 단위	**.path**	파일 시스템의 파일 또는 디렉터리.
범위 단위	**.scope**	외부에서 생성된 프로세스.
슬라이스 단위	**.slice**	시스템 프로세스를 관리하는 계층적으로 구성된 단위 그룹입니다.
소켓 단위	**.socket**	프로세스 간 통신 소켓입니다.
스왑 장치	**.swap**	스왑 장치 또는 스왑 파일.
타이머 장치	**.timer**	systemd 타이머.

12.2. SYSTEMD 주요 기능

***systemd** 시스템 및 서비스 관리자는 다음과 같은 주요 기능을 제공합니다.*

- *소켓 기반 활성화 - 부팅 시 **systemd** 는 이러한 유형의 활성화를 지원하는 모든 시스템 서비스에 대해 수신 대기 소켓을 생성하고 시작된 즉시 소켓을 이러한 서비스에 전달합니다. 이를 통해 **systemd** 는 서비스를 병렬로 시작할 수 있을 뿐 아니라 전송되는 메시지를 손실하지 않고 서*

비스를 다시 시작할 수 있습니다. 해당 소켓은 계속 액세스할 수 있으며 모든 메시지가 대기열에 있습니다.

systemd 는 소켓 기반 활성화에 소켓 유닛을 사용합니다.

- 버스 기반 활성화 - 프로세스 간 통신에 **D-Bus**를 사용하는 시스템 서비스는 클라이언트 애플리케이션이 처음으로 통신하려고 할 때 온디맨드로 시작할 수 있습니다. **systemd** 는 버스 기반 활성화를 위해 **D-Bus** 서비스 파일을 사용합니다.

- 장치 기반 활성화 - 특정 유형의 하드웨어가 연결되거나 사용 가능하게 되면 장치 기반 활성화를 지원하는 시스템 서비스를 필요에 따라 시작할 수 있습니다. **systemd** 는 장치 기반 활성화를 위해 장치 장치를 사용합니다.

- 경로 기반 활성화 - 특정 파일 또는 디렉토리가 상태를 변경할 때 경로 기반 활성화를 지원하는 시스템 서비스는 경로 기반 활성화를 시작할 수 있습니다. **systemd** 는 경로 기반 활성화를 위해 경로 단위를 사용합니다.

- 마운트 및 자동 마운트 지점 관리 - **systemd** 모니터링 및 마운트 지점 관리. **systemd** 는 마운트 지점에 마운트 단위 와 자동 마운트 지점에 마운트 단위를 사용합니다.

- 적극적인 병렬화 - 소켓 기반 활성화를 사용하기 때문에 **systemd** 는 모든 수신 대기 소켓이 있는 즉시 시스템 서비스를 병렬로 시작할 수 있습니다. 온 디맨드 활성화를 지원하는 시스템 서비스와 함께 병렬 활성화는 시스템 부팅에 필요한 시간을 크게 줄입니다.

- 트랜잭션 단위 활성화 논리 - 단위를 활성화하거나 비활성화하기 전에 **systemd** 는 종속성을 계산하고 임시 트랜잭션을 생성하고 이 트랜잭션이 일관되게 유지되는지 확인합니다. 트랜잭션이 일치하지 않는 경우 **systemd** 는 오류를 보고하기 전에 자동으로 오류를 수정하고 필수가 아닌 작업을 제거합니다.

- **SysV init**과의 역호환성 - **systemd** 는 **Linux Standard Base Core Specification** 에 설명된 **SysV init** 스크립트를 지원하여 **systemd** 서비스 유닛으로의 업그레이드 경로를 쉽게 지원합니다.

12.3. 호환성 변경

systemd 시스템 및 서비스 관리자는 **SysV init** 및 **Upstart**와 대부분 호환되도록 설계되었습니다. 다음은 **SysV init**을 사용하는 **Red Hat Enterprise Linux 6** 시스템과 관련하여 주요 호환성 변경 사항입니다.

- *systemd 에는 런레벨에 대한 지원만 제한됩니다. 이러한 런레벨에 직접 매핑할 수 있고 호환성을 이유로 이전 런레벨 명령과 함께 배포될 수 있는 여러 대상 유닛을 제공합니다. 모든 systemd 대상을 런레벨에 직접 매핑할 수 있는 것은 아니며 결과적으로 이 명령은 알 수 없는 런레벨을 나타내기 위해 N 을 반환할 수 있습니다. 가능한 경우 런레벨 명령을 사용하지 않는 것이 좋습니다.*
  systemd 대상 및 실행 수준 비교에 대한 자세한 내용은 systemd 대상 작업을 참조하십시오.

- *systemctl 유틸리티는 사용자 지정 명령을 지원하지 않습니다. start,stop 및 status 와 같은 표준 명령 외에도 SysV init 스크립트 작성자는 추가 기능을 제공하기 위해 다수의 임의 명령에 대한 지원을 구현할 수 있었습니다. 예를 들어, panic 명령을 사용하여 iptables 의 init 스크립트를 실행할 수 있습니다. 이 명령은 패닉 모드를 활성화하고 들어오고 나가는 모든 패킷을 삭제하도록 시스템을 재구성했습니다. 이는 systemd 에서 지원되지 않으며 systemctl 은 문서화된 명령만 허용합니다.*

- *systemctl 유틸리티는 systemd 에서 시작하지 않은 서비스와 통신하지 않습니다. systemd 는 시스템 서비스를 시작하면 기본 프로세스의 ID를 계속 추적하기 위해 저장합니다. systemctl 유틸리티는 이 PID를 사용하여 서비스를 쿼리하고 관리합니다. 따라서 사용자가 명령줄에서 직접 특정 데몬을 시작하는 경우 systemctl 은 현재 상태를 확인하거나 중지할 수 없습니다.*

- *systemd 는 실행 중인 서비스만 중지합니다. 이전 버전에서는 종료 시퀀스가 시작될 때 Red Hat Enterprise Linux 6 및 시스템의 이전 릴리스에서는 /etc/rc0.d/ 디렉터리에 있는 심볼릭 링크를 사용하여 상태와 관계없이 사용 가능한 모든 시스템 서비스를 중지했습니다. systemd 를 사용하면 종료 시 실행 중인 서비스만 중지됩니다.*

- *시스템 서비스는 표준 입력 스트림에서 읽을 수 없습니다. systemd 는 서비스를 시작하면 표준 입력을 /dev/null 에 연결하여 사용자와의 상호 작용을 방지합니다.*

- *시스템 서비스는 호출된 사용자 및 해당 세션에서 어떠한 컨텍스트(예: HOME 및 PATH 환경 변수)를 상속하지 않습니다. 각 서비스는 명확한 실행 컨텍스트에서 실행됩니다.*

- *SysV init 스크립트를 로드할 때 systemd 는 Linux Standard Base(LSB) 헤더로 인코딩된 종속성 정보를 읽고 런타임에 해석합니다.*

- *서비스 유닛에 대한 모든 작업에는 기본 시간 초과가 5분으로 되어 시스템이 중단되는 것을 방지할 수 있습니다. 이 값은 initscripts에서 생성된 서비스에 하드 코딩되며 변경할 수 없습니다. 그러나 개별 구성 파일을 사용하여 서비스당 시간 초과 값을 지정할 수 있습니다. 시간 제한 제한 변경을 참조하십시오.*

13장. SYSTEMCTL을 사용하여 시스템 서비스 관리

systemctl 유틸리티는 시스템 서비스를 관리하는 데 도움이 됩니다. **systemctl** 유틸리티를 사용하여 서비스 시작, 중지, 다시 시작, 활성화 및 비활성화, 서비스 나열, 시스템 서비스 상태 표시와 같은 다양한 서비스와 관련된 다양한 작업을 수행할 수 있습니다.

이 섹션에서는 **systemctl** 유틸리티를 사용하여 시스템 서비스를 관리하는 방법을 설명합니다.

13.1. SYSTEMCTL을 사용한 서비스 단위 관리

서비스 단위는 시스템의 서비스 및 데몬 상태를 제어하는 데 도움이 됩니다.

서비스 단위는 . **service** 파일 확장자로 끝납니다(예: **nfs-server.service**). 그러나 명령에서 서비스 파일 이름을 사용하는 동안 파일 확장자를 생략할 수 있습니다. **systemctl** 유틸리티는 인수가 서비스 단위라고 가정합니다. 예를 들어 **nfs-server.service** 를 중지하려면 다음 명령을 입력합니다.

```
# systemctl stop nfs-server
```

또한 일부 서비스 유닛에는 별칭 이름이 있습니다. 별칭은 단위보다 짧을 수 있으며 실제 단위 이름 대신 사용할 수 있습니다.

특정 유닛에 사용할 수 있는 모든 별칭을 찾으려면 다음을 사용합니다.

```
# systemctl show nfs-server.service -p Names
```

추가 리소스

- **systemctl**과 서비스 유틸리티 비교
- 시스템 서비스 나열
- 시스템 서비스 상태 표시
- 시스템 서비스 시작

- *시스템 서비스 다시 시작*
- *시스템 서비스 활성화*
- *시스템 서비스 비활성화*
- *시스템 서비스 중지*

13.2. SYSTEMCTL과 서비스 유틸리티 비교

*이 섹션에서는 서비스 유틸리티와 **systemctl** 명령 사용 간의 비교를 보여줍니다.*

표 13.1. systemctl과 서비스 유틸리티 비교

service	systemctl	설명
service *<name>* 시작	**systemctl start *<name>*.service**	서비스를 시작합니다.
service *<name>* stop	**systemctl stop *<name>*.service**	서비스를 중지합니다.
service *<name>* 재시작	**systemctl restart *<name>*.service**	서비스를 다시 시작합니다.
service *<name>* condrestart	**systemctl try-restart *<name>*.service**	실행 중인 경우에만 서비스를 다시 시작합니다.
service *<name>* reload	**systemctl reload *<name>*.service**	구성을 다시 로드합니다.
service *<name>* status	**systemctl status *<name>*.service** **systemctl is-active *<name>*.service**	서비스가 실행 중인지 확인합니다.
service --status-all	**systemctl list-units --type service --all**	모든 서비스의 상태를 표시합니다.

13.3. 시스템 서비스 나열

현재 로드된 모든 서비스 유닛과 사용 가능한 모든 서비스 유닛의 상태를 나열할 수 있습니다.

절차

- 현재 로드된 서비스 유닛을 모두 나열하려면 다음을 입력합니다.

```
$ systemctl list-units --type service
UNIT                    LOAD   ACTIVE SUB     DESCRIPTION
abrt-ccpp.service       loaded active exited  Install ABRT coredump hook
abrt-oops.service       loaded active running ABRT kernel log watcher
abrtd.service           loaded active running ABRT Automated Bug Reporting Tool
----
systemd-vconsole-setup.service loaded active exited  Setup Virtual Console
tog-pegasus.service            loaded active running OpenPegasus CIM Server

LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB    = The low-level unit activation state, values depend on unit type.

46 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'
```

기본적으로 **systemctl list-units** 명령은 활성 유닛만 표시합니다. 각 서비스 유닛 파일에 대해 명령은 다음을 표시합니다.

  - **UNIT:** 전체 이름
  - **LOAD:** 유닛 파일이 로드되었는지 여부
  - **EgressIP \ SUB:** 상위 레벨 및 낮은 수준의 단위 파일 활성화 상태
  - 설명: 간단한 설명

- 상태에 관계없이 로드된 유닛을 모두 나열하려면 **--all** 또는 **-a** 명령줄 옵션을 사용하여 다음 명령을 입력합니다.

```
$ systemctl list-units --type service --all
```

- 사용 가능한 모든 서비스 장치의상태(활성화/비활성화됨)를 나열하려면 다음을 입력합니다.

```
$ systemctl list-unit-files --type service
UNIT FILE                              STATE
abrt-ccpp.service                        enabled
abrt-oops.service                        enabled
abrtd.service                          enabled
...
wpa_supplicant.service                       disabled
ypbind.service                         disabled

208 unit files listed.
```

각 서비스 유닛에 대해 이 명령은 다음을 표시합니다.

  - 파일 이름 : 전체 이름

  - **STATE:** 서비스 유닛이 활성화 또는 비활성화되었는지 여부

추가 리소스

- 시스템 서비스 상태 표시

13.4. 시스템 서비스 상태 표시

서비스 유닛을 검사하여 자세한 정보를 가져오고 활성화되어 있는지 또는 실행 중인지 서비스 상태를 확인할 수 있습니다. 또한 특정 서비스 유닛 이후 또는 이전에 주문된 서비스를 볼 수도 있습니다.

절차

- 시스템 서비스에 해당하는 서비스 유닛에 대한 자세한 정보를 표시하려면 다음을 입력합니다.

```
$ systemctl status <name>.service
```

<name> 을 검사할 서비스 유닛의 이름으로 바꿉니다(예: **gdm**).

이 명령은 선택한 서비스 유닛의 이름, 짧은 설명, 사용 가능한 서비스 단위 정보에 설명된 하나 이상의 필드, **root** 사용자가 실행하는 경우, 최근 로그 항목을 표시합니다.

표 13.2. 사용 가능한 서비스 단위 정보

필드	설명
loaded	서비스 유닛이 로드되었는지 여부, 단위 파일의 절대 경로 및 장치가 활성화되었는지 여부를 나타냅니다.
active	서비스 유닛이 실행 중인지 여부 및 타임스탬프를 제공합니다.
Main PID	해당 시스템 서비스의 PID와 해당 이름.
상태	해당 시스템 서비스에 대한 추가 정보입니다.
process	관련 프로세스에 대한 추가 정보.
cgroup	관련 제어 그룹(**cgroups**)에 대한 추가 정보.

예 13.1. 서비스 상태 표시

GNOME Display Manager의 서비스 단위는 gdm.service 라고 합니다. 이 서비스 장치의 현재 상태를 확인하려면 쉘 프롬프트에서 다음을 입력합니다.

```
# systemctl status gdm.service
gdm.service - GNOME Display Manager
   Loaded: loaded (/usr/lib/systemd/system/gdm.service; enabled)
   Active: active (running) since Thu 2013-10-17 17:31:23 CEST; 5min ago
 Main PID: 1029 (gdm)
   CGroup: /system.slice/gdm.service
           ├─1029 /usr/sbin/gdm
           ├─1037 /usr/libexec/gdm-simple-slave --display-id /org/gno...
           └─1047 /usr/bin/Xorg :0 -background none -verbose -auth /r...

Oct 17 17:31:23 localhost systemd[1]: Started GNOME Display Manager.
```

- *특정 서비스 장치가 실행 중인지 확인하려면 다음을 입력합니다.*

```
$ systemctl is-active <name>.service
```

- *특정 서비스 유닛이 활성화되었는지 확인하려면 다음을 입력합니다.*

```
$ systemctl is-enabled <name>.service
```

참고

systemctl is-active 및 **systemctl is-enabled** 는 모두 지정된 서비스 장치가 실행 중이거나 활성화되어 있는 경우 종료 상태 **0** 을 반환합니다.

- 지정된 서비스 유닛 전에 시작하도록 정렬된 서비스를 확인하려면 다음을 입력합니다.

```
# systemctl list-dependencies --after <name>.service
```

<name> 을 명령의 서비스 이름으로 바꿉니다. 예를 들어 **gdm** 전에 시작하도록 정렬된 서비스 목록을 보려면 다음을 입력합니다.

```
# systemctl list-dependencies --after gdm.service
gdm.service
├─dbus.socket
├─getty@tty1.service
├─livesys.service
├─plymouth-quit.service
├─system.slice
├─systemd-journald.socket
├─systemd-user-sessions.service
└─basic.target
[output truncated]
```

- 지정된 서비스 유닛 다음에 시작하도록 정렬된 서비스를 확인하려면 다음을 입력합니다.

```
# systemctl list-dependencies --before <name>.service
```

<name> 을 명령의 서비스 이름으로 바꿉니다. 예를 들어 **gdm** 다음에 시작하도록 정렬된 서비스 목록을 보려면 다음을 입력합니다.

```
# systemctl list-dependencies --before gdm.service
gdm.service
├─dracut-shutdown.service
├─graphical.target
│ ├─systemd-readahead-done.service
│ ├─systemd-readahead-done.timer
│ └─systemd-update-utmp-runlevel.service
└─shutdown.target
```

```
├─systemd-reboot.service
└─final.target
  └─systemd-reboot.service
```

추가 리소스

- 시스템 서비스 나열

13.5. 양수 및 음수 서비스 종속성

systemd 에서 서비스 간 양수 및 음수 종속성이 있습니다. 특정 서비스를 시작하려면 하나 이상의 다른 서비스(긍정 종속성)를 시작하거나 하나 이상의 서비스(**negative dependency**)를 중지해야 할 수 있습니다.

새 서비스를 시작하려고 하면 **systemd** 는 사용자에게 명시적인 알림 없이 모든 종속 항목을 자동으로 해결합니다. 즉, 이미 서비스를 실행하고 음수 종속성으로 다른 서비스를 시작하려고 하면 첫 번째 서비스가 자동으로 중지됩니다.

예를 들어 **postfix** 서비스를 실행 중이고 **sendmail** 서비스를 시작하려고 하면 **systemd** 는 먼저 **postfix** 를 자동으로 중지합니다. 이러한 두 서비스가 충돌하며 동일한 포트에서 실행할 수 없기 때문입니다.

추가 리소스

- 시스템 서비스 시작

13.6. 시스템 서비스 시작

start 명령을 사용하여 현재 세션에서 시스템 서비스를 시작할 수 있습니다. 서비스를 시작하면 운영 체제 상태에 영향을 미칠 수 있으므로 루트 액세스 권한이 있어야 합니다.

절차

- 시스템 서비스에 해당하는 선택한 서비스 장치를 시작하려면 **root** 로 다음 명령을 입력합니다.

```
# systemctl start <name>.service
```

<name> 을 시작하려는 서비스 유닛의 이름으로 바꿉니다(예: httpd.service).

> *예 13.2. httpd.service 시작*
>
> *Apache HTTP 서버의 서비스 단위는 httpd.service 라고 합니다. 이 서비스 장치를 활성화하고 현재 세션에서 httpd 데몬을 시작하려면 root 로 다음 명령을 입력합니다.*
>
> ```
> # systemctl start httpd.service
> ```

추가 리소스

- *양수 및 음수 서비스 종속 항목*
- *시스템 서비스 활성화*
- *시스템 서비스 상태 표시*

13.7. 시스템 서비스 중지

stop 명령을 사용하여 현재 세션에서 시스템 서비스를 중지할 수 있습니다. 서비스를 중지하면 운영 체제 상태에 영향을 미칠 수 있으므로 루트 액세스 권한이 있어야 합니다.

절차

- *시스템 서비스에 해당하는 서비스 장치를 중지하려면 root 로 다음 명령을 입력합니다.*

  ```
  # systemctl stop <name>.service
  ```

  중지하려는 서비스 유닛의 이름(예: bluetooth)으로 <name> 을 바꿉니다.

> *예 13.3. bluetoothd.service 중지*
>
> *bluetoothd 데몬의 서비스 단위 이름은 bluetooth.service 입니다. 이 서비스 유닛을 비활성화하고 현재 세션에서 bluetoothd 데몬을 중지하려면 root 로 다음 명령을 입력합니다.*

```
# systemctl stop bluetooth.service
```

추가 리소스

- 시스템 서비스 비활성화
- 시스템 서비스 상태 표시

13.8. 시스템 서비스 재시작

restart 명령을 사용하여 현재 세션에서 시스템 서비스를 다시 시작할 수 있습니다. 서비스를 다시 시작하면 운영 체제 상태에 영향을 미칠 수 있으므로 루트 액세스 권한이 있어야 합니다.

다음 절차에서는 다음을 수행하는 방법을 설명합니다.

- 현재 세션에서 선택한 서비스 유닛을 중지하고 즉시 다시 시작합니다.
- 해당 서비스가 이미 실행 중인 경우에만 서비스 유닛을 재시작합니다.
- 실행을 중단하지 않고 시스템 서비스 구성 다시 로드

절차

- 시스템 서비스에 해당하는 서비스 장치를 다시 시작하려면 **root** 로 다음 명령을 입력합니다.

```
# systemctl restart <name>.service
```

<name> 을 재시작할 서비스 유닛의 이름으로 바꿉니다(예: **httpd**).

참고

선택한 서비스 장치가 실행되고 있지 않으면 이 명령도 시작합니다.

- 또는 해당 서비스가 이미 실행 중인 경우에만 서비스 장치를 다시 시작하려면 **root** 로 다음 명령을 입력합니다.

```
# systemctl try-restart <name>.service
```

- 서비스 실행을 중단하지 않고 구성을 다시 로드하려면 **root** 로 다음 명령을 입력합니다.

```
# systemctl reload <name>.service
```

참고

이 기능을 지원하지 않는 시스템 서비스는 이 명령을 무시합니다. 이러한 서비스를 다시 시작하려면 **reload-or-restart** 및 **reload-or-try-restart** 명령을 대신 사용합니다.

예 13.4. httpd.service 다시 로드

사용자가 불필요한 오류 메시지나 부분적으로 렌더링된 웹 페이지가 발생하지 않도록 하려면 **Apache HTTP Server**를 사용하여 구성을 다시 시작하고 적극적으로 처리된 요청을 중단하지 않고도 해당 구성을 편집하고 다시 로드할 수 있습니다. 이렇게 하려면 다음을 **root** 로 입력합니다.

```
# systemctl reload httpd.service
```

추가 리소스

- 시스템 서비스 상태 표시

13.9. 시스템 서비스 활성화

시스템 부팅 시 자동으로 시작하도록 서비스를 구성할 수 있습니다. **enable** 명령은 선택한 서비스 장치의 **[Install]** 섹션을 읽고 **/etc/ systemd/system/** 디렉터리 및 해당 하위 디렉터리의 **/usr/lib/systemd/system/이름.service** 파일에 적절한 심볼릭 링크를 만듭니다. 그러나 이미 존재하는 링크는 다시 작성하지 않습니다.

절차

- 부팅 시 자동으로 시스템 서비스에 해당하는 서비스 장치를 구성하려면 **root** 로 다음 명령을

입력합니다.

```
# systemctl enable <name>.service
```

<name> 을 활성화할 서비스 유닛의 이름으로 바꿉니다(예: **httpd**).

- 심볼릭 링크가 다시 생성되었는지 확인하려면 **root** 로 다음 명령을 입력합니다.

```
# systemctl reenable <name>.service
```

이 명령은 선택한 서비스 장치를 비활성화하고 즉시 다시 활성화합니다.

예 13.5. httpd.service 활성화

부팅 시 자동으로 시작하도록 **Apache HTTP Server**를 구성하려면 **root** 로 다음 명령을 입력합니다.

```
# systemctl enable httpd.service
Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to
/usr/lib/systemd/system/httpd.service.
```

추가 리소스

- 시스템 서비스 상태 표시
- 시스템 서비스 시작

13.10. 시스템 서비스 비활성화

부팅 시 서비스 장치가 자동으로 시작되지 않도록 할 수 있습니다. **disable** 명령은 선택한 서비스 장치의 **[Install]** 섹션을 읽고 **/etc/ systemd/system/** 디렉터리 및 해당 하위 디렉터리의 **/usr/lib/systemd/system/**이름**.service** 파일에 대한 적절한 심볼릭 링크를 제거합니다.

절차

- 부팅 시 자동으로 시작되지 않는 시스템 서비스에 해당하는 서비스 장치를 구성하려면 **root** 로 다음 명령을 입력합니다.

```
# systemctl disable <name>.service
```

<name> 을 비활성화하려는 서비스 유닛의 이름으로 바꿉니다(예: bluetooth).

예 13.6. bluetoothd.service 비활성화

bluetoothd 데몬의 서비스 단위 이름은 bluetooth.service 입니다. 이 서비스 장치가 부팅 시 시작되지 않도록 하려면 root 로 다음 명령을 입력합니다.

```
# systemctl disable bluetooth.service
Removed symlink /etc/systemd/system/bluetooth.target.wants/bluetooth.service.
Removed symlink /etc/systemd/system/dbus-org.bluez.service.
```

- *서비스 유닛을 마스킹하고 수동으로 또는 다른 서비스에 의해 시작되지 않도록 하려면 root 로 다음 명령을 입력합니다.*

```
# systemctl mask <name>.service
```

이 명령은 /etc/systemd/system/name.service 파일을 /dev/null 에 대한 심볼릭 링크로 대체하여 systemd 에 액세스할 수 없는 실제 장치 파일을 렌더링합니다.

- *이 작업을 취소하고 서비스 유닛을 마스크 해제하려면 다음을 입력합니다.*

```
# systemctl unmask <name>.service
```

추가 리소스

- *시스템 서비스 상태 표시*
- *시스템 서비스 중지*

14장. SYSTEMD 대상 작업

systemd 대상은 대상 단위로 표시됩니다. 대상 단위 파일은 **.target** 파일 확장자로 종료되고 유일한 목적은 종속 항목 체인을 통해 다른 **systemd** 유닛을 함께 그룹화하는 것입니다. 예를 들어 그래픽 세션을 시작하는 데 사용되는 **graphical.target** 장치 는 **GNOME Display Manager (gdm.service)** 또는 계정 서비스 **(accounts-daemon.service)** 와 같은 시스템 서비스를 시작하고, 다중 사용자**.target** 유닛 을 활성화합니다. 마찬가지로 **multi-user.target** 장치는 **NetworkManager (NetworkManager.service)** 또는 **D-Bus (dbus.service)** 와 같은 다른 필수 시스템 서비스를 시작하고 **basic.target**이라는 다른 대상 장치를 활성화합니다.

이 섹션에는 **systemd** 대상을 사용하는 동안 구현하는 절차가 포함되어 있습니다.

14.1. SYSV 실행 수준과 SYSTEMD 대상 간의 차이점

이전 버전의 **Red Hat Enterprise Linux**는 **SysV init** 또는 **Upstart**와 함께 배포되었으며 특정 작업 모드를 나타내는 사전 정의된 실행 수준 세트를 구현했습니다. 이러한 실행 수준으로 인해 **0**에서 **6**까지 번호가 지정되었으며 시스템 관리자가 특정 실행 수준을 사용하도록 설정한 경우 실행할 시스템 서비스에 의해 정의되었습니다. **Red Hat Enterprise Linux 7**부터는 실행 수준의 개념이 **systemd** 대상으로 교체되었습니다.

Red Hat Enterprise Linux 7은 이전 릴리스의 표준 실행 수준 세트와 비슷하거나 더 적은 사전 정의된 대상과 함께 배포되었습니다. 호환성을 위해 **SysV** 실행 수준에 직접 매핑되는 이러한 대상에 대한 별칭도 제공합니다.

다음 표에서는 **SysV** 실행 수준 및 해당 **systemd** 대상의 전체 목록을 제공합니다.

표 **14.1. systemd** 대상과 **SysV** 실행 수준 비교

실행 수준	대상 단위	설명
0	**runlevel0.target**, **poweroff.target**	시스템을 종료하고 전원을 끕니다.
1	**runlevel1.target**, **rescue.target**	복구 쉘을 설정합니다.
2	**runlevel2.target**, **multi-user.target**	그래픽이 아닌 다중 사용자 시스템을 설정합니다.
3	**runlevel3.target**, **multi-user.target**	그래픽이 아닌 다중 사용자 시스템을 설정합니다.

실행 수준	대상 단위	설명
4	**runlevel4.target**, **multi-user.target**	그래픽이 아닌 다중 사용자 시스템을 설정합니다.
5	**runlevel5.target**, **graphical.target**	그래픽 다중 사용자 시스템을 설정합니다.
6	**runlevel6.target**, **reboot.target**	시스템을 종료하고 재부팅합니다.

*다음 표에서는 **SysV init** 명령을 **systemctl**과 비교합니다. **systemctl** 유틸리티를 사용하여 **systemd** 대상을 확인, 변경 또는 구성합니다.*

중요

*실행 수준 및 **telinit** 명령은 계속 시스템에서 사용할 수 있으며 예상대로 작동하지만 호환성상의 이유로만 포함되어 있어야 합니다.*

표 14.2. systemctl과 SysV init 명령 비교

이전 명령	새 명령	설명
실행 수준	**systemctl list-units --type 대상**	현재 로드된 대상 유닛을 나열합니다.
telinit *실행 수준*	**systemctl isolate** ***name*.target**	현재 대상을 변경합니다.

추가 리소스

- ***man sysv init***
- ***man upstart init***
- ***man systemctl***

14.2. 기본 대상 보기

*기본 대상 단위는 **/etc/systemd/system/default.target** 파일로 표시됩니다.*

절차

- *기본적으로 사용되는 대상 단위를 확인하려면 다음을 수행합니다.*

```
$ systemctl get-default
graphical.target
```

- *심볼릭 링크를 사용하여 기본 대상을 결정하려면 다음을 수행합니다.*

```
$  ls -l /usr/lib/systemd/system/default.target
```

= 대상 유닛 보기

*기본적으로 **systemctl list-units** 명령은 활성 유닛만 표시합니다.*

절차

- *상태에 관계없이 로드된 모든 단위를 나열하려면 다음을 수행합니다.*

```
$ systemctl list-units --type target --all
```

- *현재 로드된 모든 대상 유닛을 나열하려면 다음을 수행합니다.*

```
$ systemctl list-units --type target

UNIT                   LOAD   ACTIVE SUB    DESCRIPTION
basic.target           loaded active active Basic System
cryptsetup.target      loaded active active Encrypted Volumes
getty.target           loaded active active Login Prompts
graphical.target       loaded active active Graphical Interface
local-fs-pre.target    loaded active active Local File Systems (Pre)
local-fs.target        loaded active active Local File Systems
multi-user.target      loaded active active Multi-User System
network.target         loaded active active Network
paths.target           loaded active active Paths
remote-fs.target       loaded active active Remote File Systems
sockets.target         loaded active active Sockets
sound.target           loaded active active Sound Card
spice-vdagentd.target loaded active active Agent daemon for Spice guests
```

```
swap.target           loaded active active Swap
sysinit.target        loaded active active System Initialization
time-sync.target      loaded active active System Time Synchronized
timers.target         loaded active active Timers

LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB    = The low-level unit activation state, values depend on unit type.

17 loaded units listed.
```

14.2.1. 기본 대상 변경

기본 대상 단위는 **/etc/systemd/system/default.target** 파일로 표시됩니다. 다음 절차에서는 **systemctl** 명령을 사용하여 기본 대상을 변경하는 방법을 설명합니다.

절차

1. 기본 대상 단위를 결정하려면 다음을 수행합니다.

```
# systemctl get-default
```

2. 기본적으로 다른 대상 장치를 사용하도록 시스템을 구성하려면 다음을 수행합니다.

```
# systemctl set-default multi-user.target
rm /etc/systemd/system/default.target
ln -s /usr/lib/systemd/system/multi-user.target /etc/systemd/system/default.target
```

이 명령은 **/etc/systemd/system/default.target** 파일을 **/usr/lib/systemd/system/name.target** 에 대한 심볼릭 링크로 교체합니다. 여기서 **name**은 사용하려는 대상 장치의 이름입니다. **multi-user** 를 기본적으로 사용하려는 대상 유닛의 이름으로 바꿉니다.

표 **14.3. set-default** 명령의 일반적인 대상

Basic	기본 부팅을 다루는 단위 대상
rescue	기본 시스템에서 가져오고 복구 쉘을 생성하는 단위 대상
multi-user	다중 사용자 시스템을 설정하기 위한 단위 대상
graphical	그래픽 로그인 화면을 설정하는 단위 대상

emergency	기본 콘솔에서 긴급 쉘을 시작하는 단위 대상
sysinit	시스템 초기화에 필요한 서비스를 가져오는 단위 대상

3. *재부팅*

```
# reboot
```

추가 리소스

- *대상 단위에 대한 자세한 내용은 **systemd.special manpage**를 참조하십시오.*
- *부팅 도움말 페이지*

14.2.2. 심볼릭 링크를 사용하여 기본 대상 변경

다음 절차에서는 대상에 대한 심볼릭 링크를 만들어 기본 대상을 변경하는 방법을 설명합니다.

절차

1. *기본 대상 단위를 확인합니다.*

```
#  ls -l /etc/systemd/system/default.target
```

*특정 경우에는 **/etc/systemd/system/default.target** 링크가 존재하지 않을 수 있으며 **systemd**는 **/usr** 에서 기본 대상 단위를 찾습니다. 이 경우 다음 명령을 사용하여 기본 대상 단위를 결정합니다.*

```
#  ls -l /usr/lib/systemd/system/default.target
```

2. ***/etc/systemd/system/default.target** 링크를 삭제합니다.*

```
# rm /etc/systemd/system/default.target
```

3. *심볼릭 링크를 만듭니다.*

```
#  ln -sf /usr/lib/systemd/system/graphical.target /etc/systemd/system/default.target
```

4. 시스템을 재부팅합니다.

```
# reboot
```

검증 단계

- 새로 생성된 **default.target**을 확인합니다.

```
$ systemctl get-default
multi-user.target
```

14.2.3. 현재 대상 변경

다음 절차에서는 **systemctl** 명령을 사용하여 현재 세션의 대상 유닛을 변경하는 방법을 설명합니다.

절차

- 현재 세션에서 다른 대상 유닛으로 변경하려면 다음을 수행합니다.

```
# systemctl isolate multi-user.target
```

이 명령은 **multi-user** 및 모든 종속 유닛이라는 대상 유닛을 시작하고 다른 모든 장치를 즉시 중지합니다.

multi-user 를 기본적으로 사용하려는 대상 유닛의 이름으로 바꿉니다.

검증 단계

- 새로 생성된 **default.target**을 확인합니다.

```
$ systemctl get-default
multi-user.target
```

= 복구 모드로 부팅

복구 모드는 편리한 단일 사용자 환경을 제공하며 일반적인 부팅 프로세스를 완료할 수 없는 상황에서 시스템을 복구할 수 있습니다. 복구 모드에서는 시스템이 모든 로컬 파일 시스템을 마운트하고 일부 중요한 시스템 서비스를 시작하려고 하지만 네트워크 인터페이스를 활성화하지 않거나 더 많은 사용자가 시스템에 로그인할 수 있도록 합니다.

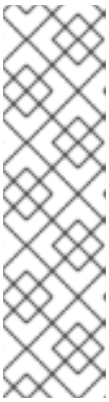
절차

- *현재 대상을 변경하고 현재 세션에 복구 모드로 전환하려면 다음을 수행합니다.*

```
# systemctl rescue

Broadcast message from root@localhost on pts/0 (Fri 2013-10-25 18:23:15 CEST):

The system is going down to rescue mode NOW!
```



참고

*이 명령은 **systemctl isolate rescue.target** 과 유사하지만 현재 시스템에 로그인한 모든 사용자에게 정보 메시지를 보냅니다.*

systemd** 가 메시지를 보내지 못하도록 하려면 **--no-wall** 명령줄 옵션을 사용하여 다음 명령을 실행합니다. **# systemctl --no-wall rescue

14.2.3.1. 긴급 모드로 부팅

긴급 모드는 가능한 가장 최소한의 환경을 제공하며 시스템이 복구 모드로 전환되지 않은 경우에도 시스템을 복구할 수 있습니다. 긴급 모드에서는 읽기용으로만 루트 파일 시스템을 마운트하고 다른 로컬 파일 시스템을 마운트하지 않고 네트워크 인터페이스를 활성화하지 않으며 몇 가지 필수 서비스만 시작합니다.

절차

- *현재 대상을 변경하고 긴급 모드로 전환하려면 다음을 수행합니다.*

```
# systemctl emergency
```

참고

*이 명령은 **systemctl isolate emergency.target** 과 유사하지만 현재 시스템에 로그인한 모든 사용자에게 정보 메시지를 보냅니다.*

systemd**가 이 메시지를 전송하지 못하도록 하려면 **--no-wall** 명령줄 옵션을 사용하여 다음 명령을 실행합니다. **# systemctl --no-wall emergency

15장. 시스템 종료, 일시 중지 및 절전 관리

이 섹션에는 운영 체제 종료, 일시 중지 또는 절전 관리에 대한 지침이 포함되어 있습니다.

15.1. 시스템 종료

시스템을 종료하려면 **systemctl** 유틸리티를 직접 사용하거나 **shutdown** 명령을 통해 이 유틸리티를 호출할 수 있습니다.

shutdown 명령을 사용할 때의 장점은 다음과 같습니다.

- 시간 인수 지원

  이는 예약된 유지 관리에 특히 유용합니다. 또한 사용자는 시스템 종료가 계획되었다는 경고에 반응할 시간이 더 많습니다.

- 종료를 취소할 수 있는 옵션

추가 리소스

- **shutdown** 명령을 사용하여 시스템 종료
- **systemctl** 명령을 사용하여 시스템 종료
- **systemctl**을 사용한 전원 관리 명령 개요

15.2. SHUTDOWN 명령을 사용하여 시스템 종료

이 절차에 따라 **shutdown** 명령을 사용하여 다양한 작업을 수행할 수 있습니다. 시스템을 종료하고 특정 시점에서 시스템의 전원을 끄거나 시스템 전원을 끄지 않고 시스템을 중지하거나 시스템 종료를 취소할 수 있습니다.

사전 요구 사항

- root 사용자로 전환합니다.

절차

- 시스템을 종료하고 특정 시점에서 시스템의 전원을 끄려면 다음 형식으로 명령을 사용합니다.

```
shutdown --poweroff hh:mm
```

여기서 hh:mm 는 24시간 시계 형식의 시간입니다. 새 로그인을 방지하기 위해 시스템을 종료하기 전에 /run/nologin 파일이 5분 후에 생성됩니다.

시간 인수를 사용하면 옵션인 wall 메시지를 명령에 추가할 수 있습니다.

또는 시스템 전원을 끄지 않고 잠시 후 시스템을 종료하고 중지하려면 다음을 사용합니다.

```
shutdown --halt +m
```

여기서 +m 은 지연 시간(분)입니다. now 키워드는 +0 의 별칭입니다.

보류 중인 종료를 취소하려면 다음을 사용합니다.

```
shutdown -c
```

추가 리소스

- shutdown(8) 매뉴얼 페이지
- systemctl 명령을 사용하여 시스템 종료

15.3. SYSTEMCTL 명령을 사용하여 시스템 종료

이 절차에 따라 systemctl 명령을 사용하여 다양한 작업을 수행할 수 있습니다. 시스템을 종료하고 시스템의 전원을 끄거나 시스템 전원을 끄지 않고 시스템을 종료하고 중지할 수 있습니다.

사전 요구 사항

- ***root** 사용자로 전환합니다.*

절차

- *시스템을 종료하고 시스템 전원을 끄려면 다음 형식으로 명령을 사용합니다.*

```
systemctl poweroff
```

또는 시스템의 전원을 끄지 않고 시스템을 종료하고 중지하려면 다음을 사용합니다.

```
systemctl halt
```

참고

*기본적으로 이러한 명령 중 하나를 실행하면 **systemd**에서 현재 시스템에 로그인한 모든 사용자에게 정보 메시지를 보냅니다. **systemd**가 이 메시지를 전송하지 못하도록 하려면 **--no-wall** 명령줄 옵션을 사용하여 선택한 명령을 실행합니다.*

추가 리소스

- ***shutdown** 명령을 사용하여 시스템 종료*

15.4. 시스템을 다시 시작

이 절차에 따라 시스템을 다시 시작할 수 있습니다.

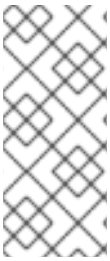
사전 요구 사항

- ***root** 사용자로 전환합니다.*

절차

- *시스템을 다시 시작하려면 다음 명령을 실행합니다.*

```
systemctl reboot
```



참고

기본적으로 이 명령을 실행하면 **systemd**에서 현재 시스템에 로그인한 모든 사용자에게 정보 메시지를 보냅니다. **systemd**가 이 메시지를 전송하지 못하도록 하려면 **--no-wall** 명령줄 옵션을 사용하여 이 명령을 실행합니다.

15.5. 시스템 일시 중지

이 절차에 따라 시스템을 일시 중단할 수 있습니다.

사전 요구 사항

- **root** 사용자로 전환합니다.

절차

- 시스템을 일시 중지하려면 다음 명령을 실행합니다.

```
systemctl suspend
```

이 명령은 시스템 상태를 **RAM**에 저장하고 **RAM** 모듈을 제외하고 시스템의 대부분의 장치의 전원을 끕니다. 시스템을 다시 켜면 시스템을 다시 부팅할 필요 없이 **RAM**에서 해당 상태를 복원합니다.

시스템 상태가 하드 디스크가 아닌 **RAM**에 저장되기 때문에 시스템을 일시 중지 모드로 복원하는 것은 절전 모드보다 훨씬 빠릅니다. 그러나 일시 중단된 시스템 상태도 정전에 취약합니다.

추가 리소스

- 시스템 최대독화

15.6. 시스템 HIBERNATING

이 절차에 따라 시스템을 최대로 지정할 수 있습니다. 즉, **system**을 켜고 시스템을 일시 중단할 수 있습니다.

사전 요구 사항

- *root 사용자로 전환합니다.*

절차

- *시스템을 최신 상태로 전환하려면 다음 명령을 실행합니다.*

```
systemctl hibernate
```

이 명령은 시스템 상태를 하드 디스크 드라이브에 저장하고 시스템의 전원을 끕니다. 시스템을 다시 켜면 시스템을 다시 부팅하지 않고도 저장된 데이터에서 해당 상태를 복원합니다.

시스템 상태가 하드 디스크에 저장되고 RAM에 저장되지 않으므로 시스템은 RAM 모듈에 전력을 공급할 필요가 없습니다. 그러나 결과적으로 시스템 장애 조치(hibernation)를 복원하는 것은 일시 중지 모드에서 복원하는 것보다 훨씬 느려집니다.

또는 RuntimeClass로 시스템을 일시 중단하려면 다음 명령을 실행합니다.

```
systemctl hybrid-sleep
```

추가 리소스

- *시스템 일시 중단*

15.7. SYSTEMCTL을 사용하여 전원 관리 명령 개요

다음 systemctl 명령 목록을 사용하여 시스템의 전원 관리를 제어할 수 있습니다.

표 15.1. systemctl 전원 관리 명령 개요

systemctl 명령	설명
systemctl halt	시스템을 중지합니다.
systemctl poweroff	시스템의 전원을 끕니다.
systemctl reboot	시스템을 다시 시작합니다.
systemctl suspend	시스템을 일시 중지합니다.

systemctl 명령	설명
systemctlloadbalancing	Hibernates the system을 실행합니다.
systemctl hybrid-sleep	Hibernates를 사용하여 시스템을 일시중지합니다.

16장. SYSTEMD 장치 파일 작업

이 장에서는 **systemd** 유닛 파일에 대한 설명을 포함합니다. 다음 섹션에서는 다음을 수행하는 방법을 보여줍니다.

- 사용자 정의 단위 파일 만들기
- **SysV init** 스크립트를 유닛 파일로 변환
- 기존 유닛 파일 수정
- 인스턴스화 단위 작업

16.1. 단위 파일 소개

유닛 파일에는 단위를 설명하고 해당 동작을 정의하는 구성 지시문이 포함되어 있습니다. 백그라운드에서 여러 개의 **systemctl** 명령이 유닛 파일을 사용하여 작동합니다. 시스템 관리자가 수동으로 단위 파일을 편집하거나 생성하려면 장치 파일을 수동으로 편집하거나 생성해야 합니다. *systemd 장치 파일에*는 시스템에 단위 파일이 저장되는 **3**개의 기본 디렉토리가 나열되며, **/etc/systemd/system/** 디렉터리는 시스템 관리자가 만들거나 사용자 지정하는 장치 파일에 대해 예약되어 있습니다.

단위 파일 이름은 다음과 같습니다.

> ***unit_name.type_extension***

여기에서 **unit_name** 은 유닛의 이름을 나타내고 **type_extension** 은 유닛 유형을 식별합니다. 장치 유형 전체 목록은 *systemd 장치 파일을*참조하십시오.

예를 들어 일반적으로 시스템에 **sshd.service** 및 **sshd.socket** 장치가 있습니다.

추가 구성 파일을 위해 디렉터리로 장치 파일을 추가할 수 있습니다. 예를 들어 **sshd.service** 에 사용자 지정 구성 옵션을 추가하려면 **sshd.service.d/custom.conf** 파일을 생성하고 추가 지시문을 삽입합니다. 구성 디렉터리에 대한 자세한 내용은 *기존 장치 파일 수정을* 참조하십시오.

또한 **sshd.service.wants/** 및 **sshd.service.requires/** 디렉터리를 생성할 수 있습니다. 이러한 디렉터리에는 **sshd** 서비스의 종속 항목인 단위 파일에 대한 심볼릭 링크가 포함되어 있습니다. 심볼릭 링크는 **[Install]** 장치 파일 옵션 또는 **[Unit]** 옵션에 따라 런타임 시 설치 중에 자동으로 생성됩니다. 이러한 디렉터리와 심볼릭 링크를 수동으로 생성할 수도 있습니다. **[Install]** 및 **[Unit]** 옵션에 대한 자세한 내용은 아래 표를 참조하십시오.

많은 단위 파일 옵션은 장치 파일이 로드될 때 장치 매개 변수로 동적으로 대체되는 와일드카드 문자열이라는 **so called** 단위 쿼리자를 사용하여 설정할 수 있습니다. 이렇게 하면 인스턴스화된 단위를 생성하기 위한 템플릿으로 사용되는 일반 장치 파일을 생성할 수 있습니다. 인스턴스화 단위 작업을 참조하십시오.

16.2. 단위 파일 구조

단위 파일은 일반적으로 다음 세 섹션으로 구성됩니다.

- **[Unit]** 섹션은 단위 유형에 의존하지 않는 일반 옵션이 포함되어 있습니다. 이러한 옵션은 단위 설명을 제공하고, 단위의 동작을 지정하고, 종속성을 다른 단위로 설정합니다. 자주 사용하는 **[Unit]** 옵션 목록은 중요 **[Unit]** 섹션 옵션을 참조하십시오.

- **[Unit type]** 섹션 - 단위에 유형별 지시문이 있는 경우 단위 유형 뒤에 이름이 지정된 섹션 아래에 그룹화됩니다. 예를 들어 서비스 장치 파일에는 **[Service]** 섹션이 포함됩니다.

- **[Install]** 섹션 - **systemctl enable** 및 **disable** 명령에서 사용하는 유닛 설치에 대한 정보를 포함합니다. **[Install]** 섹션의 옵션 목록은 **Important [Install]** 섹션 옵션을 참조하십시오.

추가 리소스

- 중요 **[Unit]** 섹션 옵션

- 중요한 [서비스] 섹션 옵션

- 중요 **[Install]** 섹션 옵션

16.3. 중요 [UNIT] 섹션 옵션

다음 표에는 **[Unit]** 섹션의 중요한 옵션이 나열되어 있습니다.

표 16.1. 중요 [Unit] 섹션 옵션

옵션 [a]	설명
설명	단위에 대한 의미 있는 설명입니다. 이 텍스트는 예를 들어 **systemctl status** 명령 출력에 표시됩니다.
문서	단위에 대한 URI 참조 설명서 목록을 제공합니다.
이후[b]	유닛이 시작되는 순서를 정의합니다. 이 단위는 **After** 에 지정된 단위가 활성화된 후에만 시작됩니다. **Requires** 와 달리,**After** 는 지정된 단위를 명시적으로 활성화하지 않습니다. **Before** 옵션은 다음과 같은 기능을 **수행합니다**.
필수 항목	다른 유닛에 대한 종속성을 구성합니다. **Requires** 에 나열된 단위는 장치와 함께 활성화됩니다. 필요한 유닛 중 하나라도 시작되지 않으면 장치가 활성화되지 않습니다.
원하는 경우	**Requires** 보다 약한 종속성을 설정합니다. 나열된 유닛 중 하나라도 성공적으로 시작되지 않으면 장치 활성화에 영향을 미치지 않습니다. 사용자 지정 유닛 종속성을 설정하는 것이 좋습니다.
충돌	**Requires** 와 반대로 음수 종속성을 설정합니다.

[a] [Unit] 섹션에서 구성할 수 있는 전체 옵션 목록은 **systemd.unit(5)** 매뉴얼 페이지를 참조하십시오.

[b] 대부분의 경우 **After** 및 **Before** 단위 파일 옵션을 사용하여 순서가 지정된 종속성만 설정하는 것으로 충분합니다. 또한 **Wants** (recommended) 또는 **Requires** 를 사용하여 요구 종속성을 설정하는 경우에도 순서 종속성을 지정해야 합니다. 이는 순서 및 요구 사항 종속성이 서로 독립적으로 작동하기 때문입니다.

16.4. 중요 [SERVICE] 섹션 옵션

다음 표에는 [Service] 섹션의 중요한 옵션이 나열되어 있습니다.

표 16.2. 중요 [Service] 섹션 옵션

옵션 [a]	설명

옵션 [a]	설명
유형	**ExecStart** 및 관련 옵션의 기능에 영향을 주는 유닛 프로세스 시작 유형을 구성합니다. 다음 중 하나: * **simple** - 기본값입니다. **ExecStart** 로 시작하는 프로세스는 서비스의 주요 프로세스입니다. * **예약** - **ExecStart** 로 시작하는 프로세스는 서비스의 주요 프로세스가 되는 하위 프로세스를 생성합니다. 상위 프로세스는 시작이 완료되면 종료됩니다. * **shot** - 이 유형은 **simple** 과 유사하지만 consequent 장치를 시작하기 전에 프로세스가 종료됩니다. * **dbus** - 이 유형은 **단순** 과 유사하지만, 주 프로세스에서 D-Bus 이름을 얻은 후에만 시작됩니다. **알림** - 이 유형은 **단순** 과 유사하지만 연속 단위는 sd_notify() 함수를 통해 알림 메시지를 보낸 후에만 시작됩니다. * **idle** - **단순** 과 유사하게 모든 작업이 완료될 때까지 서비스 바이너리의 실제 실행이 지연되어 상태 출력을 서비스의 쉘 출력과 혼합하지 않습니다.
ExecStart	단위가 시작될 때 실행할 명령 또는 스크립트를 지정합니다. **ExecStart Pre** 및 **ExecStartPost** 이전 및 이후 실행할 사용자 지정 명령을 지정합니다. **type=oneshot** 을 사용하면 순차적으로 실행되는 여러 사용자 지정 명령을 지정할 수 있습니다.
ExecStop	장치가 중지될 때 실행할 명령 또는 스크립트를 지정합니다.
ExecReload	장치가 다시 로드될 때 실행할 명령 또는 스크립트를 지정합니다.
재시작	이 옵션을 활성화하면 **systemctl** 명령의 명확한 중지를 제외하고 프로세스가 종료된 후 서비스가 다시 시작됩니다.
RemainAfterExit	True로 설정하면 모든 프로세스가 종료된 경우에도 서비스가 활성 상태로 간주됩니다. 기본값은 False입니다. 이 옵션은 **Type=oneshot** 이 구성된 경우 특히 유용합니다.

[a] [Service] 섹션에서 구성 가능한 전체 옵션 목록은 **systemd.service(5)** 매뉴얼 페이지를 참조하십시오.

16.5. 중요 [설치] 섹션 옵션

다음 표에는 [Install] 섹션의 중요한 옵션이 나열되어 있습니다.

표 16.3. 중요 [설치] 섹션 옵션

옵션 [a]	설명
별칭	은 단원에 대해 공백으로 구분된 추가 이름 목록을 제공합니다. **systemctl enable** 을 제외한 대부분의 systemctl 명령은 실제 유닛 이름 대신 별칭을 사용할 수 있습니다.
RequiredBy	단위에 종속된 유닛 목록입니다. 이 유닛을 활성화하면 **RequiredBy** 에 나열된 단위가 단위에 대한 종속성이 요구됩니다.
WantedBy	장치가 약하게 의존하는 단위 목록입니다. 이 유닛을 활성화하면 **WantedBy** 에 나열된 단위가 단위에 대해 **Want** dependency를 얻습니다.
Also	유닛을 설치하거나 제거할 유닛 목록을 지정합니다.
DefaultInstance	인스턴스화 단위로 제한되는 이 옵션은 장치가 활성화된 기본 인스턴스를 지정합니다. 인스턴스화 단위 작업을 참조하십시오.
[a] [Install] 섹션에서 구성 가능한 전체 옵션 목록은 **systemd.unit(5)** 매뉴얼 페이지를 참조하십시오.	

16.6. 사용자 지정 유닛 파일 생성

*단위 파일을 처음부터 생성하는 데는 몇 가지 사용 사례가 있습니다. 사용자 지정 데몬을 실행하고, **sshd** 서비스의 두 번째 인스턴스를 사용하여 사용자 지정 장치 파일 생성에서 몇 가지 기존 서비스의 두 번째 인스턴스를만들 수 있습니다.*

반면 기존 단위의 동작을 수정하거나 확장하려는 경우 기존 장치 파일 수정 에서 명령을 사용하십시오.

절차

다음 절차에서는 사용자 지정 서비스를 생성하는 일반적인 프로세스를 설명합니다.

1.

사용자 지정 서비스로 실행 파일을 준비합니다. 사용자 지정 스크립트 또는 소프트웨어 공급자가 제공하는 실행 파일일 수 있습니다. 필요한 경우 사용자 지정 서비스의 기본 프로세스에 대해 상수 PID를 유지하도록 PID 파일을 준비합니다. 서비스에 대한 셸 변수를 저장할 환경 파일을 포함할 수도 있습니다. 소스 스크립트가 실행 가능하고(chmod a+x) 대화형이 아닌지 확인합니다.

2. */etc/systemd/system/ 디렉터리에 유닛 파일을 만들고 올바른 파일 권한이 있는지 확인합니다. root 로 실행:*

```
touch /etc/systemd/system/name.service

chmod 664 /etc/systemd/system/name.service
```

name 을 생성할 서비스 이름으로 바꿉니다. 파일은 실행 파일일 필요가 없습니다.

3. *이전 단계에서 만든이름.service 파일을 열고 서비스 구성 옵션을 추가합니다. 생성하려는 서비스 유형에 따라 사용할 수 있는 다양한 옵션이 있습니다. 단위 파일 구조를 참조하십시오.*

다음은 네트워크 관련 서비스의 유닛 구성 예입니다.

```
[Unit]
Description=service_description
After=network.target

[Service]
ExecStart=path_to_executable
Type=forking
PIDFile=path_to_pidfile

[Install]
WantedBy=default.target
```

다음과 같습니다.

- *service_description 은 저널 로그 파일과 systemctl status 명령의 출력에 표시되는 정보적 설명입니다.*

- *after 설정 은 네트워크가 실행된 후에만 서비스가 시작되는지 확인합니다. 다른 관련 서비스 또는 대상의 공백으로 구분된 목록을 추가합니다.*

- *path_to_executable* 은 실제 실행 파일의 경로를 나타냅니다.

- *type=forking* 은 *fork* 시스템 호출을 수행하는 데몬에 사용됩니다. 서비스의 기본 프로세스는 *path_to_pidfile* 에 지정된 *PID*를 사용하여 생성됩니다. 중요한 [서비스] 섹션에서 다른 시작 유형을 찾습니다.

- *WantedBy* 는 서비스를 시작해야 하는 대상 또는 대상을 지정합니다. 이러한 대상을 실행 수준의 이전 개념을 대체하는 것으로 생각하십시오.

4. *root* 로 다음 명령을 실행하여 새이름*.service* 파일이 있음을 *systemd* 에 알립니다.

```
systemctl daemon-reload

systemctl start name.service
```

주의

새 장치 파일을 생성하거나 기존 유닛 파일을 수정한 후 항상 *systemctl daemon-reload* 명령을 실행합니다. 그러지 않으면 디스크의 *systemd* 및 실제 서비스 유닛 파일의 상태가 일치하지 않아 *systemctl start* 또는 *systemctl enable* 명령이 실패할 수 있었습니다. 참고: 많은 단위가 있는 시스템에서 이 작업은 각 단위의 상태를 직렬화하고 나중에 다시 로드하는 동안 역직렬화해야 하므로 시간이 오래 걸릴 수 있습니다.

16.7. SSHD 서비스의 두 번째 인스턴스를 사용하여 사용자 지정 유닛 파일 생성

시스템 관리자는 서비스 인스턴스를 여러 개 구성하고 실행해야 하는 경우가 많습니다. 이 작업은 원래 서비스 구성 파일의 사본을 생성하고 서비스의 기본 인스턴스와 충돌하지 않도록 특정 매개 변수를 수정하여 수행됩니다. 다음 절차에서는 *sshd* 서비스의 두 번째 인스턴스를 생성하는 방법을 보여줍니다.

절차

1. 두 번째 데몬에서 사용할 *sshd_config* 파일의 사본을 만듭니다.

```
# cp /etc/ssh/sshd{,-second}_config
```

2. 이전 단계에서 생성한 **sshd-second_config** 파일을 편집하여 다른 포트 번호 및 **PID** 파일을 두 번째 데몬에 할당합니다.

```
Port 22220
PidFile /var/run/sshd-second.pid
```

Port 및 **PidFile** 옵션에 대한 자세한 내용은 **sshd_config(5)** 매뉴얼 페이지를 참조하십시오. 선택한 포트가 다른 서비스에서 사용되지 않는지 확인합니다. **PID** 파일은 서비스를 실행하기 전에 존재할 필요가 없으며, 서비스 시작 시 자동으로 생성됩니다.

3. **sshd** 서비스에 대한 **systemd** 장치 파일의 사본을 생성합니다.

```
# cp /usr/lib/systemd/system/sshd.service /etc/systemd/system/sshd-second.service
```

4. 이전 단계에서 생성한 **sshd-second.service** 를 다음과 같이 변경합니다.

   a. **Description** 옵션을 수정합니다.

   ```
   Description=OpenSSH server second instance daemon
   ```

   b. 첫 번째 인스턴스가 이미 시작된 후에만 시작하도록 **After** 옵션에 지정된 서비스에 **sshd.service**를 추가합니다.

   ```
   After=syslog.target network.target auditd.service sshd.service
   ```

   c. **sshd**의 첫 번째 인스턴스에는 키 생성이 포함되어 있으므로 **ExecStartPre=/usr/sbin/sshd-keygen** 행을 제거합니다.

   d. 대체 구성 파일을 사용하도록 **-f /etc/ssh/sshd-second_config** 매개변수를 **sshd** 명령에 추가합니다.

   ```
   ExecStart=/usr/sbin/sshd -D -f /etc/ssh/sshd-second_config $OPTIONS
   ```

   e. 위의 수정 후 **sshd-second.service**는 다음과 같이 표시됩니다.

   ```
   [Unit]
   ```

```
Description=OpenSSH server second instance daemon
After=syslog.target network.target auditd.service sshd.service

[Service]
EnvironmentFile=/etc/sysconfig/sshd
ExecStart=/usr/sbin/sshd -D -f /etc/ssh/sshd-second_config $OPTIONS
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process
Restart=on-failure
RestartSec=42s

[Install]
WantedBy=multi-user.target
```

5. SELinux를 사용하는 경우 sshd의 두 번째 인스턴스에 대한 포트를 SSH 포트에 추가합니다. 그렇지 않으면 sshd의 두 번째 인스턴스가 포트에 바인딩하도록 거부됩니다.

```
# semanage port -a -t ssh_port_t -p tcp 22220
```

6. 부팅 시 자동으로 시작되도록 sshd-second.service를 활성화합니다.

```
# systemctl enable sshd-second.service
```

7. systemctl status 명령을 사용하여 sshd-second.service가 실행 중인지 확인합니다.

8. 서비스에 연결하여 포트가 올바르게 활성화되어 있는지 확인합니다.

```
$ ssh -p 22220 user@server
```

방화벽이 사용 중인 경우 sshd의 두 번째 인스턴스에 대한 연결을 허용하도록 적절하게 구성되었는지 확인합니다.

16.8. SYSV INIT 스크립트를 단위 파일로 변환

SysV init 스크립트를 유닛 파일로 변환하는 데 시간이 걸리기 전에 변환이 아직 다른 곳에서 수행되지 않았는지 확인하십시오. Red Hat Enterprise Linux에 설치된 모든 핵심 서비스는 기본 장치 파일과 함께 제공되며 여러 타사 소프트웨어 패키지에 동일하게 적용됩니다.

init 스크립트를 유닛 파일로 변환하려면 스크립트를 분석하고 필요한 정보를 추출해야 합니다. 이 데이터를 기반으로 단위 파일을 만들 수 있습니다. init 스크립트는 서비스 유형에 따라 크게 달라질 수 있으므

로 이 장에 설명된 것보다 번역에 더 많은 구성 옵션을 사용해야 할 수 있습니다. **init** 스크립트로 사용 가능한 일부 수준의 사용자 지정은 더 이상 **systemd** 장치에서 지원되지 않습니다.

변환에 필요한 대부분의 정보는 스크립트의 헤더에 제공됩니다. 다음 예제에서는 **Red Hat Enterprise Linux 6**에서 **postfix** 서비스를 시작하는 데 사용되는 **init** 스크립트의 열기 섹션을 보여줍니다.

```
#!/bin/bash
# postfix      Postfix Mail Transfer Agent
# chkconfig: 2345 80 30
# description: Postfix is a Mail Transport Agent, which is the program that moves mail from
one machine to another.
# processname: master
# pidfile: /var/spool/postfix/pid/master.pid
# config: /etc/postfix/main.cf
# config: /etc/postfix/master.cf
### BEGIN INIT INFO
# Provides: postfix MTA
# Required-Start: $local_fs $network $remote_fs
# Required-Stop: $local_fs $network $remote_fs
# Default-Start: 2 3 4 5
# Default-Stop: 0 1 6
# Short-Description: start and stop postfix
# Description: Postfix is a Mail Transport Agent, which is the program that moves mail from
one machine to another.
### END INIT INFO
```

위의 예에서 **# chkconfig** 및 **#** 설명으로 시작하는 행만 필수이므로 다른 **init** 파일에서 나머지를 찾지 못할 수 있습니다. **BEGIN INIT INFO** 와 **END INIT INFO** 행 사이에 있는 텍스트는 **Linux Standard Base(LSB)** 헤더 라고 합니다. 지정된 경우 **LSB** 헤더에는 서비스 설명, 종속 항목 및 기본 실행 수준을 정의하는 지시문이 포함됩니다. 다음은 새 단위 파일에 필요한 데이터를 수집하는 분석 작업의 개요입니다. **postfix init** 스크립트는 예제로 사용됩니다.

16.9. SYSTEMD 서비스 설명 찾기

#description.로 시작하는 줄에서 스크립트에 대한 설명 정보를 찾을 수 있습니다. 이 설명은 단위 파일의 **[Unit]** 섹션에 있는 **Description** 옵션에 있는 서비스 이름과 함께 사용합니다. **LSB** 헤더는 **#Short-Description** 및 **#Description** 행에 유사한 데이터를 포함할 수 있습니다.

16.10. SYSTEMD 서비스 종속 항목 찾기

LSB 헤더에는 서비스 간 종속성을 형성하는 여러 지시문이 포함될 수 있습니다. 대부분 **systemd** 장치 옵션으로 번역할 수 있습니다. 다음 표를 참조하십시오.

표 **16.4. LSB** 헤더의 종속성 옵션

LSB 옵션	설명	단위 파일 Equivalent
제공	다른 init 스크립트 ($" 접두사 사용)에서 참조할 수 있는 서비스의 부팅 기능 이름을 지정합니다. 유닛 파일이 파일 이름별로 다른 유닛을 참조하면 이 작업이 더 이상 필요하지 않습니다.	-
required-Start	필요한 서비스의 부팅 기능 이름이 포함되어 있습니다. 이는 순서 종속성으로 변환되며 부팅 기능 이름은 해당 서비스 또는 해당 서비스의 유닛 파일 이름으로 교체됩니다. 예를 들어 **postfix** 의 경우 $network에 대한 Required-Start 종속성이 network.target에 대한 After dependency로 변환됩니다.	**후,전에**
should-Start	Required-Start보다 약한 종속성을 구성합니다. Should-Start 종속성은 서비스 시작에 영향을 미치지 않습니다.	**후,전에**
required-Stop,Should-Stop	음수 종속성을 구성합니다.	**충돌**

16.11. 서비스의 기본 대상 찾기

***#chkconfig** 로 시작하는 줄에는 세 개의 숫자 값이 포함되어 있습니다. 가장 중요한 것은 서비스가 시작되는 기본 실행 수준을 나타내는 첫 번째 번호입니다. 이러한 실행 수준을 동등한 **systemd** 대상에 매핑합니다. 그런 다음 해당 대상을 장치 파일의 **[Install]** 섹션에 있는 **WantedBy** 옵션에 나열합니다. 예를 들어 **postfix** 는 이전에 다중 사용자.**target** 및 **graphical.target**으로 변환되는 실행 수준 **2, 3, 4** 및 **5**에서 시작되었습니다. **graphical.target**은 **journalctl.target**에 따라 달라지므로 둘 다 지정할 필요는 없습니다. 또한 **LSB** 헤더에서 **#Default-Start** 및 **#Default-Stop** 행에 **default** 및 금지된 실행 수준 정보에 대한 정보를 찾을 수 있습니다.*

***#chkconfig** 행에 지정된 다른 두 값은 **init** 스크립트의 시작 및 종료 우선 순위를 나타냅니다. 이러한 값은 **init** 스크립트를 로드하는 경우 **systemd** 에서 해석되지만 동일한 유닛 파일이 없습니다.*

16.12. 서비스에서 사용하는 파일 찾기

***init** 스크립트는 전용 디렉터리에서 함수 라이브러리를 로드해야 하며 구성, 환경 및 **PID** 파일을 가져올 수 있습니다. 환경 변수는 **init** 스크립트 헤더에서 **#config** 로 시작하는 행에서 지정되며 **EnvironmentFile** 장치 파일 옵션으로 변환됩니다. **#pidfile init** 스크립트 행에 지정된 **PID** 파일은 **PIDFile** 옵션을 사용하여 유닛 파일로 가져옵니다.*

init 스크립트 헤더에 포함되지 않은 주요 정보는 서비스 실행 파일의 경로이며 서비스에 필요할 가능성이 있는 기타 파일입니다. 이전 버전의 Red Hat Enterprise Linux에서 init 스크립트는 Bash case 문을 사용하여 start,stop, 또는 restart 와 같은 기본 작업에서 서비스 동작을 정의했습니다. postfix init 스크립트에서 발췌한 내용은 service start에서 실행할 코드 블록을 보여줍니다.

```
conf_check() {
    [ -x /usr/sbin/postfix ] || exit 5
    [ -d /etc/postfix ] || exit 6
    [ -d /var/spool/postfix ] || exit 5
}

make_aliasesdb() {
 if [ "$(/usr/sbin/postconf -h alias_database)" == "hash:/etc/aliases" ]
 then
  # /etc/aliases.db might be used by other MTA, make sure nothing
  # has touched it since our last newaliases call
  [ /etc/aliases -nt /etc/aliases.db ] ||
   [ "$ALIASESDB_STAMP" -nt /etc/aliases.db ] ||
   [ "$ALIASESDB_STAMP" -ot /etc/aliases.db ] || return
  /usr/bin/newaliases
  touch -r /etc/aliases.db "$ALIASESDB_STAMP"
 else
  /usr/bin/newaliases
 fi
}

start() {
 [ "$EUID" != "0" ] && exit 4
 # Check that networking is up.
 [ ${NETWORKING} = "no" ] && exit 1
 conf_check
 # Start daemons.
 echo -n $"Starting postfix: "
 make_aliasesdb >/dev/null 2>&1
 [ -x $CHROOT_UPDATE ] && $CHROOT_UPDATE
 /usr/sbin/postfix start 2>/dev/null 1>&2 && success || failure $"$prog start"
 RETVAL=$?
 [ $RETVAL -eq 0 ] && touch $lockfile
        echo
 return $RETVAL
}
```

init 스크립트의 확장성을 통해 start() 함수에서 호출되는 conf_check() 및 make_aliasesdb() 를 지정할 수 있습니다. 더 자세히 살펴보면 기본 서비스 실행 /usr/sbin/journal , / etc/journal/ and / var/ spool/ avoided/ 구성 디렉토리 및 /usr/sbin/ postconf/ 디렉토리 등 여러 외부 파일과 디렉터리가 언급됩니다.

systemd 는 사전 정의된 작업만 지원하지만 ExecStartPre, ExecStart Post, Exec Stop 및 Exec Reload 옵션을 사용하여 사용자 지정 실행 파일을 실행할 수 있습니다. 지원 스크립트와 함께 /usr/sbin/postfix 는 서비스 시작 시 실행됩니다. 복잡한 init 스크립트를 변환하려면 스크립트의 모든 문

용도를 이해해야 합니다. 일부 설명은 운영 체제 버전에 고유하므로 번역할 필요가 없습니다. 반면, 새로운 환경에서는 물론 서비스 실행 파일 및 지원 파일에서도 일부 조정이 필요할 수 있습니다.

16.13. 기존 장치 파일 수정

*시스템에 설치된 서비스는 **/usr/lib/systemd/system/** 디렉터리에 저장된 기본 장치 파일과 함께 제공됩니다. 시스템 관리자는 이러한 파일을 직접 수정하지 않아야 하므로 사용자 지정을 **/etc/systemd/system/** 디렉터리의 구성 파일로 제한해야 합니다.*

절차

1. *필요한 변경의 범위에 따라 다음 방법 중 하나를 선택합니다.*

   - ***/etc/systemd/system/단위.d/** 에 보조 구성 파일의 디렉터리를 만듭니다. 이 방법은 대부분의 사용 사례에 권장됩니다. 원래 장치 파일을 참조하는 동안 추가 기능을 사용하여 기본 구성을 확장할 수 있습니다. 따라서 패키지 업그레이드에 도입된 기본 단위의 변경 사항은 자동으로 적용됩니다. 자세한 내용은 기본 장치 구성 확장을 참조하십시오.*

   - ***/etc/systemd/system/** 에서 원래 장치 파일 **/usr/lib/systemd/system/** 의 복사본을 만들고 변경합니다. 복사는 원본 파일을 덮어쓰므로 패키지 업데이트로 인한 변경 사항은 적용되지 않습니다. 이 방법은 패키지 업데이트에 관계없이 유지되어야 하는 중요한 단위 변경을 수행하는 데 유용합니다. 자세한 내용은 기본 장치 구성 덮어쓰기 를 참조하십시오.*

2. *단위의 기본 구성으로 돌아가려면 **/etc/systemd/system/**.에서 사용자 지정 생성 구성 파일을 삭제합니다.*

3. *시스템을 재부팅하지 않고 장치 파일에 변경 사항을 적용하려면 다음을 실행합니다.*

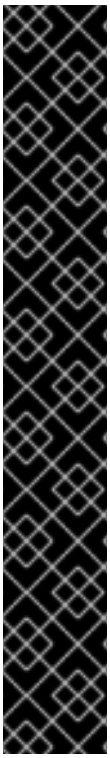
   ```
   systemctl daemon-reload
   ```

   ***daemon-reload** 옵션은 모든 유닛 파일을 다시 로드하고 전체 종속성 트리를 다시 생성합니다. 이 트리는 유닛 파일에 변경 사항을 즉시 적용하는 데 필요합니다. 또는 **root** 사용자가 실행해야 하는 다음 명령을 사용하여 동일한 결과를 얻을 수 있습니다.*

   ```
   init q
   ```

4. *수정된 유닛 파일이 실행 중인 서비스에 속하는 경우 새 설정을 수락하려면 이 서비스를 다시 시작해야 합니다.*

```
systemctl restart name.service
```



중요

SysV initscript에서 처리하는 서비스의 종속성 또는 시간 초과와 같은 속성을 수정하려면 initscript 자체를 수정하지 마십시오. 대신 다음에 설명된 대로 서비스에 대한 systemd 드롭인 구성 파일을 생성합니다. 기본 단위 구성 확장 및 기본 장치 구성 Overriding the default unit configuration.

그런 다음 이 서비스를 일반 systemd 서비스와 동일한 방식으로 관리합니다.

예를 들어 네트워크 서비스 구성을 확장하려면 /etc/rc.d/init.d/network initscript 파일을 수정하지 마십시오. 대신 새 디렉토리 /etc/systemd/system/network.service.d/ 및 systemd 드롭인 파일 /etc/systemd/system/network.service.d/my_config.conf 를 생성합니다. 그런 다음 수정된 값을 드롭인 파일에 배치합니다. 참고: systemd 는 network.service 로 네트워크 서비스를 알고 있으므로 생성된 디렉토리를 network.service.d라고 지정해야 합니다.

16.14. 기본 단위 구성 확장

이 섹션에서는 추가 구성 옵션으로 기본 유닛 파일을 확장하는 방법을 설명합니다.

절차

1. 추가 구성 옵션을 사용하여 기본 유닛 파일을 확장하려면 먼저 /etc/systemd/system/.에 구성 디렉터리를 만듭니다. 서비스 장치를 확장하는 경우 root 로 다음 명령을 실행합니다.

```
mkdir /etc/systemd/system/name.service.d/
```

name 을 확장할 서비스의 이름으로 변경합니다. 위의 구문은 모든 유닛 유형에 적용됩니다.

2. 이전 단계에서 만든 디렉터리에 구성 파일을 생성합니다. 파일 이름은 .conf 접미사로 끝나야 합니다. 유형:

```
touch /etc/systemd/system/name.service.d/config_name.conf
```

config_name 을 구성 파일의 이름으로 바꿉니다. 이 파일은 일반 단위 파일 구조를 준수하므로 적절한 섹션에서 모든 지시문을 지정해야 합니다. 단위 파일 구조를 참조하십시오.

예를 들어 사용자 지정 종속성을 추가하려면 다음 콘텐츠를 사용하여 구성 파일을 생성합니다.

```
[Unit]
Requires=new_dependency
After=new_dependency
```

*여기서 **new_dependency** 는 유닛이 종속성으로 표시됨을 나타냅니다. 또 다른 예로는 기본 프로세스가 종료된 후 서비스를 다시 시작하는 구성 파일이 **30**초로 지연됩니다.*

```
[Service]
Restart=always
RestartSec=30
```

하나의 작업에만 중점을 둔 작은 구성 파일을 생성하는 것이 좋습니다. 이러한 파일은 쉽게 이동하거나 다른 서비스의 구성 디렉토리에 연결할 수 있습니다.

3. *단위에 대한 변경 사항을 적용하려면 **root** 로 실행합니다.*

```
systemctl daemon-reload
systemctl restart name.service
```

예 16.1. httpd.service 구성 확장

*Apache 서비스를 시작할 때 사용자 정의 쉘 스크립트가 자동으로 실행되도록 **httpd.service** 유닛을 수정하려면 다음 단계를 수행합니다.*

1. *디렉터리 및 사용자 지정 구성 파일을 생성합니다.*

```
# mkdir /etc/systemd/system/httpd.service.d/
```

```
# touch /etc/systemd/system/httpd.service.d/custom_script.conf
```

2. *Apache를 사용하여 자동으로 시작하려는 스크립트가 **/usr/local/bin/custom.sh** 에 있는 경우 **custom_script.conf** 파일에 다음 텍스트를 삽입합니다.*

```
[Service]
ExecStartPost=/usr/local/bin/custom.sh
```

3. 단위 변경 사항을 적용하려면 다음을 실행합니다.

```
# systemctl daemon-reload
```

```
# systemctl restart httpd.service
```

참고

/etc/systemd/system/ 의 구성 파일은 /usr/lib/systemd/system/ 의 장치 파일보다 우선합니다. 따라서 구성 파일에 Description 또는 ExecStart 와 같이 한 번만 지정할 수 있는 옵션이 포함된 경우 이 옵션의 기본값이 재정의됩니다. systemd-delta 명령의 출력에서 재정의된 단위 모니터링 에 설명된 대로, 이러한 장치는 합계에 있는 경우에도 항상 [EXTENDED]로 표시됩니다.

16.15. 기본 장치 구성 덮어쓰기

이 섹션에서는 기본 단위 구성을 재정의하는 방법을 설명합니다.

절차

1. 장치 파일을 제공하는 패키지를 업데이트한 후 계속 변경하려면 먼저 파일을 /etc/systemd/system/ 디렉터리에 복사합니다. 이렇게 하려면 root 로 다음 명령을 실행하십시오.

```
cp /usr/lib/systemd/system/name.service /etc/systemd/system/name.service
```

여기서 name 은 수정할 서비스 유닛의 이름을 나타냅니다. 위의 구문은 모든 유닛 유형에 적용됩니다.

2. 텍스트 편집기로 복사된 파일을 열고 원하는 대로 변경합니다. 단위 변경 사항을 적용하려면 root 로 실행합니다.

```
systemctl daemon-reload
systemctl restart name.service
```

16.16. 제한 시간 제한 변경

서비스당 시간 제한 값을 지정하여 시스템 중단 서비스가 중단되는 것을 방지할 수 있습니다. 그렇지 않

으면 표준 서비스의 경우 시간 초과가 기본적으로 **90**초로 설정되고 **SysV** 호환 서비스의 경우 **300**초로 설정됩니다.

예를 들어 **httpd** 서비스의 시간 제한 제한을 확장하려면 다음을 수행합니다.

절차

1. **httpd** 장치 파일을 **/etc/systemd/system/** 디렉터리에 복사합니다.

```
cp /usr/lib/systemd/system/httpd.service /etc/systemd/system/httpd.service
```

2. **/etc/systemd/system/httpd.service** 파일을 열고 **[Service]** 섹션에서 **TimeoutStartUSec** 값을 지정합니다.

```
...
[Service]
...
PrivateTmp=true
TimeoutStartSec=10

[Install]
WantedBy=multi-user.target
...
```

3. **systemd** 데몬을 다시 로드합니다.

```
systemctl daemon-reload
```

4. 선택 사항: 새 시간 초과 값을 확인합니다.

```
systemctl show httpd -p TimeoutStartUSec
```

참고

제한 시간 제한을 전역적으로 변경하려면 **/etc/systemd/system.conf** 파일에 **DefaultTimeoutStartSec** 를 입력합니다.

16.17. 재정의된 유닛 모니터링

이 섹션에서는 재정의되거나 수정된 유닛 파일에 대한 개요를 표시하는 방법에 대해 설명합니다.

절차

- 재정의 또는 수정된 단위 파일의 개요를 표시하려면 다음 명령을 사용합니다.

```
systemd-delta
```

예를 들어 위 명령의 출력은 다음과 같습니다.

```
[EQUIVALENT] /etc/systemd/system/default.target →
/usr/lib/systemd/system/default.target
[OVERRIDDEN] /etc/systemd/system/autofs.service →
/usr/lib/systemd/system/autofs.service

--- /usr/lib/systemd/system/autofs.service      2014-10-16 21:30:39.000000000 -0400
+ /etc/systemd/system/autofs.service  2014-11-21 10:00:58.513568275 -0500
@@ -8,7 +8,8 @@
 EnvironmentFile=-/etc/sysconfig/autofs
 ExecStart=/usr/sbin/automount $OPTIONS --pid-file /run/autofs.pid
 ExecReload=/usr/bin/kill -HUP $MAINPID
-TimeoutSec=180
+TimeoutSec=240
+Restart=Always

 [Install]
 WantedBy=multi-user.target

[MASKED]    /etc/systemd/system/cups.service →
/usr/lib/systemd/system/cups.service
[EXTENDED]   /usr/lib/systemd/system/sssd.service →
/etc/systemd/system/sssd.service.d/journal.conf

4 overridden configuration files found.
```

16.18. 인스턴스화 단위 작업

런타임 시 단일 템플릿 구성 파일에서 여러 단위를 인스턴스화할 수 있습니다. "@" 문자는 템플릿을 표시하고 장치와 장치를 연결하는 데 사용됩니다. 인스턴스화된 단위는 다른 유닛 파일(**Requires** 또는 **Wants** 옵션 사용) 또는 **systemctl start** 명령으로 시작할 수 있습니다. 인스턴스화된 서비스 유닛의 이름은 다음과 같습니다.

```
template_name@instance_name.service
```

여기서 **template_name** 은 템플릿 구성 파일의 이름을 나타냅니다. **instance_name** 을 유닛 인스턴스

의 이름으로 바꿉니다. 여러 인스턴스는 장치의 모든 인스턴스에 공통된 구성 옵션을 사용하여 동일한 템플릿 파일을 가리킬 수 있습니다. 템플릿 단위 이름에는 다음과 같은 형식이 있습니다.

```
unit_name@.service
```

*예를 들어, 단위 파일의 다음 **Wants** 설정:*

```
Wants=getty@ttyA.service getty@ttyB.service
```

*먼저 **systemd**가 지정된 서비스 장치를 검색합니다. 이러한 유닛을 찾을 수 없는 경우 "@"과 유형 접미사 사이의 일부가 무시되고, **systemd** 는 **getty@.service** 파일을 검색하고, 해당 파일에서 구성을 읽고, 서비스를 시작합니다.*

*예를 들어 **getty@.service** 템플릿에는 다음 지시문이 포함되어 있습니다.*

```
[Unit]
Description=Getty on %I
…
[Service]
ExecStart=-/sbin/agetty --noclear %I $TERM
…
```

*위 템플릿에서 **getty@ttyA.service** 및 **getty@ttyB.service**가 인스턴스화되면 **Description=**은 **ttyA** 및 **Getty on tty B** 로 확인됩니다.*

16.19. 중요 단위 지정자

단위 지정자 라고 하는 와일드카드 문자를 모든 단위 구성 파일에 사용할 수 있습니다. 단위는 특정 유닛 매개변수를 대체하고 런타임 시 해석됩니다. 다음 표에는 템플릿 유닛에 특히 유용한 유닛이 나열되어 있습니다.

*표 **16.5.** 중요 단위 지정자*

단위 지정기	meaning	설명
%n	전체 단위 이름	은 유형 접미사를 포함하여 전체 단위 이름을 나타냅니다. **%N** 은 동일한 의미이지만 금지된 문자를 ASCII 코드로 대체합니다.

단위 지정기	meaning	설명
%p	접두사 이름	은 유형 접미사가 제거된 단위 이름입니다. 인스턴스화된 단위 %p는 "@" 문자 앞에 있는 단위 이름의 일부를 나타냅니다.
%i	인스턴스 이름	"@" 문자와 유형 접미사 사이의 인스턴스화된 단위 이름의 일부입니다. **%I** 는 동일한 의미이지만 ASCII 코드의 금지된 문자도 대체합니다.
%H	호스트 이름	는 장치 구성이 로드될 때 실행 중인 시스템의 호스트 이름을 나타냅니다.
%t	런타임 디렉터리	**root** 사용자의 **/run** 또는 권한이 없는 사용자에 대한 XDG_RUNTIME_DIR 변수의 값인 런타임 디렉터리를 나타냅니다.

*단위 연산자의 전체 목록은 **systemd.unit(5)** 매뉴얼 페이지를 참조하십시오.*

16.20. 추가 리소스

- *특정 서비스를 적용하는 서비스 단위 파일을 작성하는 방법*
- ***systemd** 서비스 장치 정의에 필요한 종속성을 결정하는 방법*
- *유닛 파일 작성에 대한 유용한 정보가 있습니까?*
- ***RHEL 7** 및 **systemd**에서 서비스에 대한 제한 설정 방법*

17장. SYSTEMD를 최적화하여 부팅 시간을 단축

systemd 장치 파일 목록은 기본적으로 활성화되어 있습니다. 이러한 장치 파일에 의해 정의된 시스템 서비스는 부팅 시 자동으로 실행되며 부팅 시간에 영향을 미칩니다.

이 섹션에서는 다음을 설명합니다.

- 시스템 부팅 성능을 검사하는 툴입니다.
- 기본적으로 활성화된 **systemd** 장치의 용도와 부팅 시간을 단축하기 위해 이러한 **systemd** 장치를 안전하게 비활성화할 수 있는 상황은 다음과 같습니다.

17.1. 시스템 부팅 성능 검사

시스템 부팅 성능을 검사하려면 **systemd- aoctets** 명령을 사용할 수 있습니다. 이 명령에는 사용 가능한 많은 옵션이 있습니다. 그러나 이 섹션에서는 부팅 시간을 단축하기 위해 **systemd** 튜닝에 중요할 수 있는 선택된 항목만 다룹니다.

모든 옵션에 대한 전체 목록 및 자세한 설명은 **systemd-a analyzing man** 페이지를 참조하십시오.

사전 요구 사항

- 부팅 시간을 조정하기 위해 **systemd**를 검사하기 전에 활성화된 모든 서비스를 나열할 수 있습니다.

절차

```
$ systemctl list-unit-files --state=enabled
```

전체 부팅 시간 분석

절차

- *마지막으로 성공한 부팅이 수행된 시간에 대한 전체 정보를 보려면 다음을 사용합니다.*

```
$ systemd-analyze
```

단위 초기화 시간 분석

절차

- *각 **systemd** 장치의 초기화 시간에 대한 정보는 다음을 사용합니다.*

```
$ systemd-analyze blame
```

마지막 성공적인 부팅 중에 초기화하는 데 걸리는 시간에 따라 출력에 내림차순으로 유닛이 나열됩니다.

중요 단위 확인

절차

- *마지막으로 성공한 부팅 시 초기화하는 데 가장 많은 시간이 걸리는 단위를 확인하려면 다음을 사용합니다.*

```
$ systemd-analyze critical-chain
```

출력은 빨간색 색상으로 부팅을 크게 늦추는 단위를 강조 표시합니다.

그림 17.1. systemd-aoctets critical-chain 명령의 출력

```
[admin@localhost ~]$ systemd-analyze critical-chain
The time after the unit is active or started is printed after the "@" character.
The time the unit takes to start is printed after the "+" character.

graphical.target @19.706s
└─multi-user.target @19.706s
  └─tuned.service @5.616s +3.397s
    └─network.target @5.614s
      └─wpa_supplicant.service @16.025s +125ms
        └─dbus.service @2.461s
          └─basic.target @2.444s
            └─sockets.target @2.444s
              └─iscsiuio.socket @2.444s
                └─sysinit.target @2.431s
                  └─systemd-update-utmp.service @2.419s +10ms
                    └─auditd.service @2.292s +126ms
                      └─systemd-tmpfiles-setup.service @2.228s +63ms
                        └─import-state.service @2.171s +54ms
                          └─local-fs.target @2.168s
                            └─run-user-42.mount @9.536s
                              └─local-fs-pre.target @2.112s
                                └─lvm2-monitor.service @2.087s +25ms
                                  └─dm-event.socket @968ms
                                    └─-.mount
                                      └─system.slice
                                        └─-.slice
[admin@localhost ~]$
```

17.2. 안전하게 비활성화할 수 있는 서비스를 선택하기 위한 가이드

시스템의 부팅 시간이 긴 경우 기본적으로 부팅 시 활성화된 일부 서비스를 비활성화하여 시스템의 부팅 시간을 단축할 수 있습니다.

이러한 서비스를 나열하려면 다음을 실행합니다.

```
$ systemctl list-unit-files --state=enabled
```

서비스를 비활성화하려면 다음을 실행합니다.

```
# systemctl disable service_name
```

그러나 운영 체제가 안전하고 필요한 방식으로 작동하는 방식으로 특정 서비스가 활성화되어 있어야 합니다.

*아래 표를 사용하여 안전하게 비활성화할 수 있는 서비스를 선택하는 가이드로 사용할 수 있습니다. 테이블에는 **Red Hat Enterprise Linux**의 최소 설치에서 기본적으로 활성화된 모든 서비스가 나열되며, 각 서비스에 대해 이 서비스를 안전하게 비활성화할 수 있는지 여부가 표시됩니다.*

또한 이 표에서는 서비스를 비활성화할 수 있는 상황 또는 서비스를 비활성화하지 않아야 하는 이유에 대한 추가 정보를 제공합니다.

표 17.1. RHEL의 최소 설치에서 기본적으로 활성화된 서비스

서비스 이름	Disabled 할 수 있습니까?	자세한 정보
auditd.service	제공됨	커널에서 감사 메시지가 필요하지 않은 경우에만 **auditd.service** 를 비활성화합니다. **auditd.service** 를 비활성화하면 **/var/log/audit/audit.log** 파일이 생성되지 않습니다. 그 결과 사용자 로그인, 서비스 시작 또는 암호 변경 등 몇 가지 일반적으로 검토되는 일부 작업 또는 이벤트를 소급적으로 검토할 수 없습니다. 또한 auditd에는 커널 부분과 서비스 자체의 두 가지 부분이 있습니다. **systemctl disable auditd** 명령을 사용하면 커널 부분은 아니지만 서비스만 비활성화합니다. 전체 시스템 감사를 비활성화하려면 커널 명령줄에 **audit=0** 을 설정합니다.
autovt@.service	제공되지 않음	이 서비스는 실제로 필요할 때만 실행되므로 비활성화할 필요가 없습니다.

서비스 이름	Disabled 할 수 있습니까?	자세한 정보
crond.service	제공됨	crond.service를 비활성화하면 crontab의 항목이 실행되지 않습니다.
dbus-org.fedoraproject.FirewallD1.service	제공됨	**firewalld.service**에 대한 심볼릭 링크
dbus-org.freedesktop.NetworkManager.service	제공됨	**NetworkManager.service**에 대한 심볼릭 링크
dbus-org.freedesktop.nm-dispatcher.service	제공됨	**NetworkManager-dispatcher.service**에 대한 심볼릭 링크
firewalld.service	제공됨	방화벽이 필요하지 않은 경우에만 **firewalld.service** 를 비활성화합니다.
getty@.service	제공되지 않음	이 서비스는 실제로 필요할 때만 실행되므로 비활성화할 필요가 없습니다.
import-state.service	제공됨	네트워크 스토리지에서 부팅할 필요가 없는 경우에만 **import-state.service** 를 비활성화합니다.
irqbalance.service	제공됨	CPU가 하나만 있는 경우에만 **irqbalance.service** 를 비활성화합니다. 여러 CPU가 있는 시스템에서 **irqbalance.service** 를 비활성화하지 마십시오.
kdump.service	제공됨	커널 크래시에서 보고서가 필요하지 않은 경우에만 **kdump.service** 를 비활성화합니다.
loadmodules.service	제공됨	**/etc/rc.modules 또는 /etc/sysconfig/modules** 디렉터리가 존재하지 않는 한 이 서비스는 시작되지 않습니다. 즉, 최소한의 RHEL 설치에서는 시작되지 않습니다.
lvm2-monitor.service	제공됨	LVM(Logical Volume Manager)을 사용하지 않는 경우에만 **lvm2-monitor.service** 를 비활성화합니다.
microcode.service	제공되지 않음	CPU에서 마이크로 코드 소프트웨어의 업데이트를 제공하므로 서비스를 비활성화하지 마십시오.

서비스 이름	Disabled 할 수 있습니까?	자세한 정보
NetworkManager-dispatcher.service	제공됨	네트워크 구성 변경(예: 정적 네트워크)에 대한 알림이 필요하지 않은 경우에만 **NetworkManager-dispatcher.service** 를 비활성화합니다.
NetworkManager-wait-online.service	제공됨	부팅 직후에 작동하는 네트워크 연결이 필요하지 않은 경우에만 **NetworkManager-wait-online.service** 를 비활성화합니다. 서비스가 활성화되면 시스템은 네트워크 연결이 작동하기 전에 부팅을 완료하지 않습니다. 이는 부팅 시간을 크게 연장할 수 있습니다.
NetworkManager.service	제공됨	네트워크에 연결되지 않은 경우에만 **NetworkManager.service** 를 비활성화합니다.
nis-domainname.service	제공됨	NIS(Network Information Service)를 사용하지 않는 경우에만 **nis-domainname.service** 를 비활성화합니다.
rhsmcertd.service	제공되지 않음	
rngd.service	제공됨	시스템에 엔트로피가 많이 필요하지 않거나 하드웨어 생성기가 없는 경우에만 **rngd.service** 를 비활성화합니다. 이 서비스는 X.509 인증서 생성(예: FreeIPA 서버)에 사용되는 시스템과 같이 좋은 엔트로피가 필요한 환경에서 필요합니다.
rsyslog.service	제공됨	영구 로그가 필요하지 않은 경우에만 **rsyslog.service** 를 비활성화하거나 **systemd-journald** 를 영구 모드로 설정합니다.
selinux-autorelabel-mark.service	제공됨	SELinux를 사용하지 않는 경우에만 **selinux-autorelabel-mark.service** 를 비활성화합니다.
sshd.service	제공됨	OpenSSH 서버에서 원격 로그인이 필요하지 않은 경우에만 **sshd.service** 를 비활성화합니다.
sssd.service	제공됨	네트워크를 통해 시스템에 로그인하는 사용자가 없는 경우에만 **sssd.service** 를 비활성화합니다(예: LDAP 또는 Kerberos 사용). **sssd.service** 를 비활성화하는 경우 모든 **sssd-*** 유닛을 비활성화하는 것이 좋습니다.
syslog.service	제공됨	**rsyslog.service** 의 별칭
tuned.service	제공됨	성능 튜닝을 사용해야 하는 경우에만 **tuned.service** 를 비활성화합니다.

서비스 이름	Disabled 할 수 있습니까?	자세한 정보
lvm2-lvmpolld.socket	제공됨	LVM(Logical Volume Manager)을 사용하지 않는 경우에만 **lvm2-lvmpolld.socket** 을 비활성화합니다.
dnf-makecache.timer	제공됨	패키지 메타데이터를 자동으로 업데이트할 필요가 없는 경우에만 **dnf-makecache.timer** 를 비활성화합니다.
unbound-anchor.timer	제공됨	DNSSEC(DNS 보안 확장)에 대한 루트 신뢰 앵커의 일일 업데이트가 필요하지 않은 경우에만 **unbound-anchor.timer** 를 비활성화합니다. 이 루트 신뢰 앵커는 DNSSEC 검증을 위해 Unbound resolver 및 resolver 라이브러리에서 사용합니다.

서비스에 대한 자세한 정보를 찾으려면 다음 명령 중 하나를 실행합니다.

```
$ systemctl cat <service_name>
```

```
$ systemctl help <service_name>
```

***systemctl cat** 명령은 **/usr/lib/systemd/system/<service>** 에 있는 서비스 파일의 내용과 적용 가능한 모든 재정의를 제공합니다. 해당 재정의에는 **/etc/systemd/system/<service>** 파일의 장치 파일 덮어쓰기 또는 해당 **unit.type.d** 디렉터리의 드롭인 파일이 포함됩니다.*

*드롭인 파일에 대한 자세한 내용은 **systemd.unit man** 페이지를 참조하십시오.*

***systemctl help** 명령은 특정 서비스의 도움말 페이지를 표시합니다.*

17.3. 추가 리소스

- ***systemctl(1)** 도움말 페이지*
- ***Systemd (1)** 도움말 페이지*
- ***systemd-delta(1)** 도움말 페이지*

- ***systemd.directives(7)* 도움말 페이지**
- ***systemd.unit(5)* 도움말 페이지**
- ***systemd.service(5)* 도움말 페이지**
- ***systemd.target(5)* 도움말 페이지**
- ***systemd.kill(5)* 도움말 페이지**
- *시스템 홈 페이지*

18장. 사용자 및 그룹 계정 관리 소개

사용자 및 그룹 제어는 RHEL(Red Hat Enterprise Linux) 시스템 관리의 핵심 요소입니다. 각 RHEL 사용자에게는 고유한 로그인 자격 증명이 있으며 다양한 그룹에 할당하여 시스템 권한을 사용자 지정할 수 있습니다.

18.1. 사용자 및 그룹 소개

파일을 생성하는 사용자는 해당 파일의 소유자와 해당 파일의 그룹 소유자입니다. 파일에는 소유자, 그룹 및 해당 그룹 외부의 사용자에 대한 별도의 읽기, 쓰기, 실행 권한이 할당됩니다. 파일 소유자는 root 사용자만 변경할 수 있습니다. 파일에 대한 액세스 권한은 root 사용자와 파일 소유자 모두에서 변경할 수 있습니다. 일반 사용자는 자신이 소유한 파일의 그룹 소유권을 멤버로 변경할 수 있습니다.

각 사용자는 사용자 ID(UID)라는 고유한 숫자 ID 번호와 연결됩니다. 각 그룹은 그룹 ID (GID)와 연결됩니다. 그룹 내의 사용자는 해당 그룹이 소유한 파일을 읽고, 쓰고, 실행할 수 있는 동일한 권한을 공유합니다.

18.2. 예약된 사용자 및 그룹 ID 구성

RHEL은 시스템 사용자 및 그룹에 대해 1000 미만의 사용자 및 그룹 ID를 예약합니다. 예약된 사용자 및 그룹 ID는 setup 패키지에서 찾을 수 있습니다. 예약된 사용자 및 그룹 ID를 보려면 다음을 사용합니다.

```
cat /usr/share/doc/setup*/uidgid
```

예약된 범위가 나중에 증가할 수 있으므로 5000에서 시작하는 새 사용자 및 그룹에 ID를 할당하는 것이 좋습니다.

기본적으로 새 사용자에 할당된 ID를 5000에서 시작하도록 하려면 /etc/login.defs 파일에서 UID_MIN 및 GID_MIN 매개변수를 수정합니다.

절차

새 사용자에게 할당된 ID를 수정하려면 기본적으로 5000에서 시작합니다.

1. 선택한 편집기에서 /etc/login.defs 파일을 엽니다.

2. *자동 **UID** 선택의 최소 값을 정의하는 행을 찾습니다.*

```
# Min/max values for automatic uid selection in useradd
#
UID_MIN                  1000
```

3. ***UID_MIN** 값을 **5000**에서 시작하도록 수정합니다.*

```
# Min/max values for automatic uid selection in useradd
#
UID_MIN                  5000
```

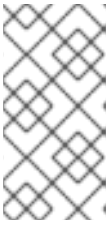
4. *자동 **GID** 선택을 위한 최소 값을 정의하는 행을 찾습니다.*

```
# Min/max values for automatic gid selection in groupadd
#
GID_MIN                  1000
```

5. ***GID_MIN** 값을 수정하여 **5000**에서 시작합니다.*

```
# Min/max values for automatic gid selection in groupadd
#
GID_MIN                  5000
```

*일반 사용자에 대해 동적으로 할당된 **UID** 및 **GID**는 이제 **5000**에서 시작합니다.*



참고

***UID_MIN** 및 **GID_MIN** 값을 변경하기 전에 생성된 **UID** 및 **GID**의 사용자 및 그룹은 변경되지 않습니다.*

*이렇게 하면 새 사용자 그룹이 **UID** 및 **GID**와 **5000+ ID**를 가질 수 있습니다.*

주의

1000 제한을 유지하는 시스템과 충돌하지 않도록 SYS_UID_MAX 를 변경하여 시스템에서 예약된 ID를 생성하지 마십시오.

18.3. 사용자 개인 그룹

RHEL은 사용자 개인 그룹 (UPG) 시스템 구성을 사용하므로 UNIX 그룹을 보다 쉽게 관리할 수 있습니다. 새 사용자가 시스템에 추가될 때마다 사용자 개인 그룹이 생성됩니다. 사용자 개인 그룹은 생성된 사용자와 동일한 이름을 가지며, 해당 사용자는 사용자 개인 그룹의 유일한 멤버입니다.

UPG는 여러 사용자 간의 프로젝트에서의 협업을 단순화합니다. 또한 UPG 시스템 구성을 사용하면 사용자 및 이 사용자가 파일 또는 디렉터리를 모두 수정할 수 있으므로 새로 생성된 파일 또는 디렉터리에 대한 기본 권한을 안전하게 설정할 수 있습니다.

모든 그룹 목록은 /etc/group 구성 파일에 저장됩니다.

19장. 웹 콘솔에서 사용자 계정 관리

RHEL 웹 콘솔은 터미널에 직접 액세스하지 않고도 광범위한 관리 작업을 실행할 수 있는 그래픽 인터페이스를 제공합니다. 예를 들어 시스템 사용자 계정을 추가, 편집 또는 제거할 수 있습니다.

이 섹션을 읽으면 다음을 알 수 있습니다.

- 기존 계정이 있는 위치에서.
- 새 계정을 추가하는 방법
- 암호 만료를 설정하는 방법
- 사용자 세션을 종료하는 방법 및 시기

사전 요구 사항

- RHEL 웹 콘솔을 설정합니다. 자세한 내용은 RHEL 웹 콘솔 사용하기 시작을 참조하십시오.
- 관리자 권한이 할당된 계정으로 RHEL 웹 콘솔에 로그인합니다. 자세한 내용은 RHEL 웹 콘솔에 로그인을 참조하십시오.

19.1. 웹 콘솔에서 관리되는 시스템 사용자 계정

RHEL 웹 콘솔에 사용자 계정이 표시되는 경우 다음을 수행할 수 있습니다.

- 시스템에 액세스할 때 사용자를 인증합니다.
- 시스템에 대한 액세스 권한을 설정합니다.

RHEL 웹 콘솔은 시스템에 있는 모든 사용자 계정을 표시합니다. 따라서 웹 콘솔에 처음 로그인한 직후 적어도 하나의 사용자 계정을 볼 수 있습니다.

RHEL 웹 콘솔에 로그인한 후 다음 작업을 수행할 수 있습니다.

- 새 사용자 계정을 생성합니다.
- 매개 변수를 변경합니다.
- 계정을 잠급니다.
- 사용자 세션을 종료합니다.

19.2. 웹 콘솔을 사용하여 새 계정 추가

다음 단계를 사용하여 시스템에 사용자 계정을 추가하고 RHEL 웹 콘솔을 통해 계정에 관리 권한을 설정합니다.

사전 요구 사항

- RHEL 웹 콘솔을 설치하고 액세스할 수 있어야 합니다. 자세한 내용은 웹 콘솔 설치를 참조하십시오.

절차

1. RHEL 웹 콘솔에 로그인합니다.
2. 계정을 클릭합니다.
3. 새 계정 만들기를 클릭합니다.

1. 전체 이름 필드에 사용자의 전체 이름을 입력합니다.

   RHEL 웹 콘솔은 전체 이름에서 사용자 이름을 자동으로 권장하고 사용자 이름 필드에 입력합

니다. 첫 번째 이름의 첫 글자와 전체 성으로 구성된 원래 이름 지정 규칙을 사용하지 않으려면 제안을 업데이트합니다.

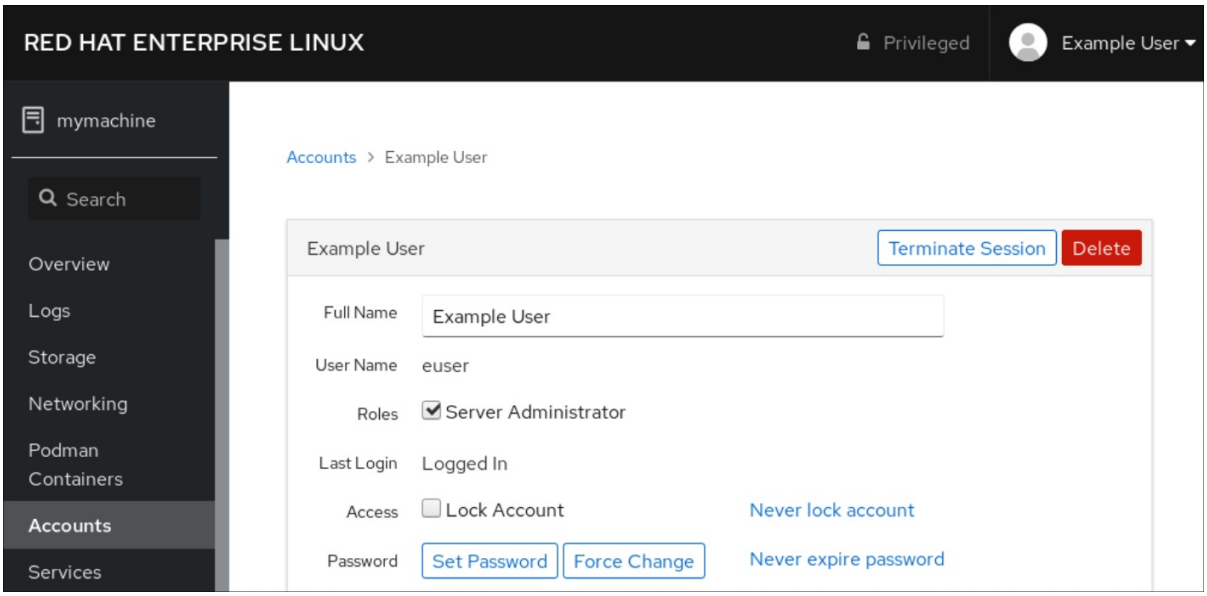
2. **Password/Confirm** 필드에 암호를 입력하고 암호가 올바른지 확인하도록 다시 입력합니다.

필드 아래에 배치된 색상 표시줄에는 암호가 약한 사용자를 생성할 수 없는 입력한 암호의 보안 수준이 표시됩니다.

1. 만들기를 클릭하여 설정을 저장하고 대화 상자를 종료합니다.

2. 새로 생성된 계정을 선택합니다.

3. **Roles** (역할) 항목에서 **Server Administrator** 를 선택합니다.



이제 계정 설정에서 새 계정을 볼 수 있으며 자격 증명을 사용하여 시스템에 연결할 수 있습니다.

19.3. 웹 콘솔에서 암호 만료 강제 적용

기본적으로 사용자 계정은 만료되지 않도록 암호를 설정합니다. 정의된 일 수 후에 만료되도록 시스템 암호를 설정할 수 있습니다. 암호가 만료되면 다음 로그인 시도에서 암호 변경을 묻는 메시지가 표시됩니

다.

절차

1. ***RHEL 9** 웹 콘솔에 로그인합니다.*

2. *계정을 클릭합니다.*

3. *암호 만료를 적용할 사용자 계정을 선택합니다.*

4. *사용자 계정 설정에서 두 번째 편집 을 클릭합니다.*

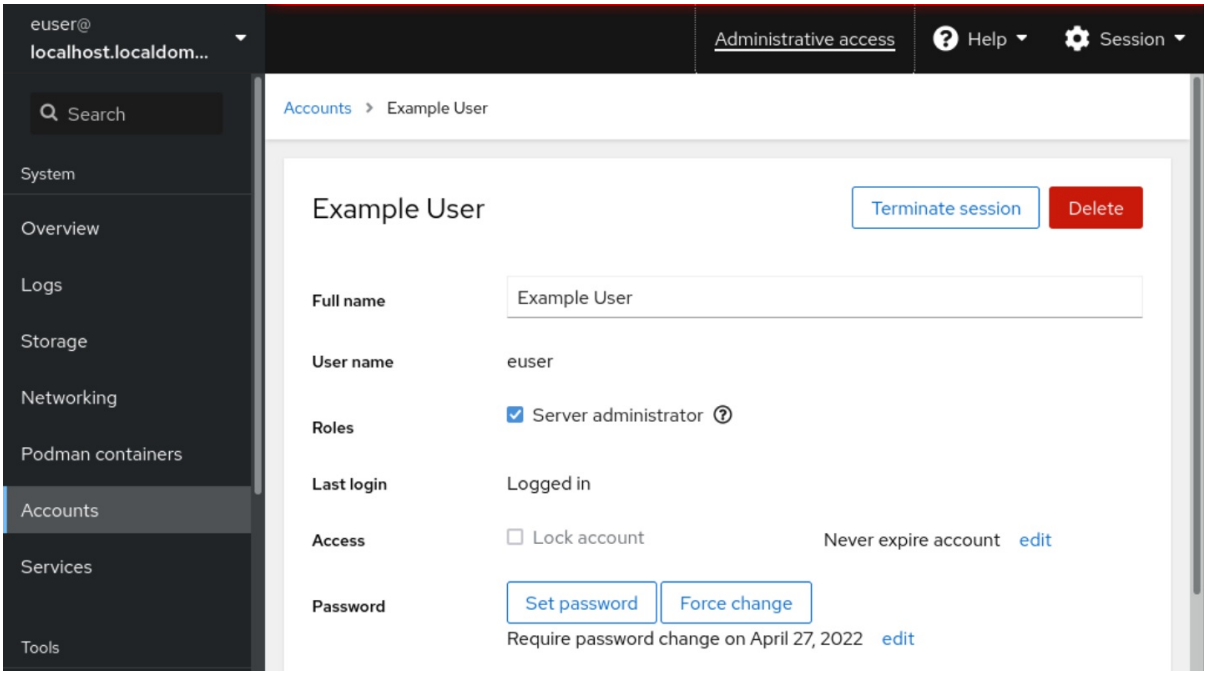
5. *암호 만료 대화 상자에서 **Require password change every ...**를 선택합니다. **days** 을(를) 입력하고 암호가 만료된 날 수를 나타내는 양의 정수 숫자를 입력합니다.*

6. *변경을 클릭합니다.*

검증 단계

- *암호 만료가 설정되었는지 확인하려면 계정 설정을 엽니다.*

  ***RHEL 9** 웹 콘솔에는 만료 날짜가 있는 링크가 표시됩니다.*



19.4. 웹 콘솔에서 사용자 세션 종료

사용자는 시스템에 로그인할 때 사용자 세션을 생성합니다. 사용자 세션을 종료하는 것은 사용자가 시스템에서 로그아웃하는 것을 의미합니다. 구성 변경에 민감한 관리 작업을 수행해야 하는 경우(예: 시스템 업그레이드) 도움이 될 수 있습니다.

***RHEL 9** 웹 콘솔의 각 사용자 계정에서 현재 사용 중인 웹 콘솔 세션을 제외하고 계정의 모든 세션을 종료할 수 있습니다. 이렇게 하면 시스템에 대한 액세스를 차단할 수 없습니다.*

절차

1. ***RHEL 9** 웹 콘솔에 로그인합니다.*
2. *계정을 클릭합니다.*
3. *세션을 종료할 사용자 계정을 클릭합니다.*
4. ***Terminate Session** 을 클릭합니다.*

   ***Terminate Session** 버튼이 비활성화된 경우 사용자는 시스템에 로그인되지 않습니다.*

***RHEL** 웹 콘솔은 세션을 종료합니다.*

20장. 명령줄에서 사용자 관리

CLI(명령줄 인터페이스)를 사용하여 사용자 및 그룹을 관리할 수 있습니다. 이를 통해 Red Hat Enterprise Linux 환경에서 사용자 및 사용자 그룹을 추가, 제거 및 수정할 수 있습니다.

20.1. 명령줄에서 새 사용자 추가

이 섹션에서는 useradd 유틸리티를 사용하여 새 사용자를 추가하는 방법을 설명합니다.

사전 요구 사항

- 루트 액세스

절차

- 새 사용자를 추가하려면 다음을 사용합니다.

```
# useradd options username
```

옵션을 useradd 명령의 명령줄 옵션으로 바꾸고 username 을 사용자 이름으로 교체합니다.

예 20.1. 새 사용자 추가

사용자 ID 5000 을 사용하여 사용자 sarah 를 추가하려면 다음을 사용하십시오.

+

```
# useradd -u 5000 sarah
```

검증 단계

- 새 사용자가 추가되었는지 확인하려면 id 유틸리티를 사용합니다.

```
# id sarah
```

출력이 반환됩니다.

```
uid=5000(sarah) gid=5000(sarah) groups=5000(sarah)
```

추가 리소스

- **useradd man** 페이지

20.2. 명령줄에서 새 그룹 추가

이 섹션에서는 **groupadd** 유틸리티를 사용하여 새 그룹을 추가하는 방법을 설명합니다.

사전 요구 사항

- 루트 액세스

절차

- 새 그룹을 추가하려면 다음을 사용합니다.

```
# groupadd options group-name
```

옵션을 **groupadd** 명령의 명령줄 옵션으로 바꾸고 **group-name** 을 그룹 이름으로 교체합니다.

예 **20.2.** 새 그룹 추가

그룹 **ID**가 **5000** 으로 그룹 **installers**를 추가하려면 다음을 사용합니다.

+

```
# groupadd -g 5000 sysadmins
```

검증 단계

- 새 그룹이 추가되었는지 확인하려면 **tail** 유틸리티를 사용합니다.

```
# tail /etc/group
```

출력이 반환됩니다.

```
sysadmins:x:5000:
```

추가 리소스

- **groupadd man** 페이지

20.3. 명령줄에서 사용자를 보조 그룹에 추가

보조 그룹에 사용자를 추가하여 권한을 관리하거나 특정 파일 또는 장치에 대한 액세스를 활성화할 수 있습니다.

사전 요구 사항

- 루트 액세스

절차

- 사용자 보조 그룹에 그룹을 추가하려면 다음을 사용합니다.

```
# usermod --append -G group-name username
```

group-name 을 그룹 이름으로 바꾸고 **username** 을 사용자 이름으로 교체합니다.

예 **20.3.** 보조 그룹에 사용자 추가

사용자 **administrator**를 **system-administrators** 그룹에 추가하려면 다음을 사용합니다.

```
# usermod --append -G system-administrators sysadmin
```

검증 단계

- *새 그룹이 사용자 **administrator**의 보조 그룹에 추가되었는지 확인하려면 다음을 사용합니다.*

```
# groups sysadmin
```

출력이 표시됩니다.

```
sysadmin : sysadmin system-administrators
```

20.4. 그룹 디렉터리 생성

***UPG** 시스템 구성에서 **set-group** 식별 권한(**set gid bit**)을 디렉터리에 적용할 수 있습니다. **setgid** 비트를 사용하면 디렉터리를 더 간단하게 공유하는 그룹 프로젝트를 관리할 수 있습니다. **setgid** 비트를 디렉터리에 적용하면 해당 디렉터리 내에서 생성된 파일이 디렉터리를 소유하는 그룹에 자동으로 할당됩니다. 이 그룹 내에서 쓰기 및 실행할 권한이 있는 사용자는 이제 디렉터리에서 파일을 생성, 수정 및 삭제할 수 있습니다.*

다음 섹션에서는 그룹 디렉터리를 만드는 방법을 설명합니다.

사전 요구 사항

- *루트 액세스*

절차

1. *디렉터리를 생성합니다.*

```
# mkdir directory-name
```

__directory-name__ 을 디렉터리 이름으로 바꿉니다.

2. *그룹을 생성합니다.*

```
# groupadd group-name
```

group-name 을 그룹 이름으로 바꿉니다.

3. 그룹에 사용자를 추가합니다.

```
# usermod --append -G group-name username
```

group-name 을 그룹 이름으로 바꾸고 **username** 을 사용자 이름으로 교체합니다.

4. 디렉터리의 사용자 및 그룹 소유권을 **group-name** 그룹과 연결합니다.

```
# chown :group-name directory-name
```

group-name 을 그룹 이름으로 바꾸고 **directory-name** 을 디렉터리 이름으로 바꿉니다.

5. 사용자가 파일 및 디렉터리를 생성 및 수정하고 **setgid** 비트를 설정하여 **directory-name** 디렉터리 내에 이 권한을 적용할 수 있도록 쓰기 권한을 설정합니다.

```
# chmod g+rwxs directory-name
```

directory-name 을 디렉터리 이름으로 바꿉니다.

이제 **group-name** 그룹의 모든 멤버는 **directory-name** 디렉터리에서 파일을 생성하고 편집할 수 있습니다. 새로 생성된 파일은 **group-name** 그룹의 그룹 소유권을 유지합니다.

검증 단계

- 설정된 권한의 정확성을 확인하려면 다음을 사용합니다.

```
# ls -ld directory-name
```

directory-name 을 디렉터리 이름으로 바꿉니다.

출력이 반환됩니다.

drwxrwsr-x. 2 root group-name 6 Nov 25 08:45 directory-name

21장. 명령줄을 사용하여 사용자 그룹 편집

사용자는 파일 및 폴더에 대한 유사한 액세스 권한을 가진 사용자의 논리적 컬렉션을 허용하는 특정 그룹 세트에 속합니다. 명령줄에서 기본 및 보조 사용자 그룹을 편집하여 사용자의 권한을 변경할 수 있습니다.

21.1. 기본 및 보조 사용자 그룹

그룹은 특정 파일에 대한 액세스 권한 부여와 같은 공통 목적을 위해 여러 사용자 계정을 함께 연결하는 엔티티입니다.

Linux에서 사용자 그룹은 기본 또는 보조 역할을 할 수 있습니다. 기본 및 보조 그룹에는 다음 속성이 있습니다.

기본 그룹

- 모든 사용자에게는 항상 하나의 기본 그룹만 있습니다.
- 사용자의 기본 그룹을 변경할 수 있습니다.

보조 그룹

- 기존 사용자를 기존 보조 그룹에 추가하여 그룹 내에서 동일한 보안 및 액세스 권한으로 사용자를 관리할 수 있습니다.
- 사용자는 0개 또는 여러 개의 보조 그룹의 구성원이 될 수 있습니다.

21.2. 사용자의 기본 및 보조 그룹 나열

사용자 그룹을 나열하여 자신이 속한 기본 및 보조 그룹을 확인할 수 있습니다.

절차

- 사용자의 기본 그룹과 보조 그룹의 이름을 표시합니다.

```
$ groups user-name
```

user-name 을 사용자 이름으로 변경합니다. 사용자 이름을 제공하지 않으면 명령은 현재 사용자의 그룹 멤버십을 표시합니다. 첫 번째 그룹은 기본 그룹 다음에 선택적 보조 그룹이 옵니다.

예 21.1. 사용자 sarah의 그룹 목록:

```
$ groups sarah
```

출력이 표시됩니다.

```
sarah : sarah wheel developer
```

User sarah 에는 기본 **group sarah** 가 있으며, 보조 그룹 **wheel** 및 **developer** 의 구성원입니다.

예 21.2. 사용자 marc의 그룹 목록:

```
$ groups marc
```

출력이 표시됩니다.

```
marc : marc
```

사용자 **marc** 에는 기본 그룹 **marc** 및 보조 그룹이 없습니다.

21.3. 사용자의 기본 그룹 변경

기존 사용자의 기본 그룹을 새 그룹으로 변경할 수 있습니다.

사전 요구 사항

1. 루트 액세스

2.

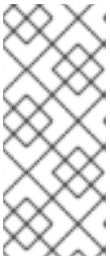
새 그룹이 존재해야 합니다.

절차

- *사용자의 기본 그룹을 변경합니다.*

```
# usermod -g group-name user-name
```

***group-name** 을 새 기본 그룹의 이름으로 바꾸고 **user-name** 을 사용자 이름으로 교체합니다.*



참고

사용자의 기본 그룹을 변경하면 명령은 사용자 홈 디렉터리에 있는 모든 파일의 그룹 소유권을 새 기본 그룹으로 자동 변경합니다. 사용자의 홈 디렉터리 외부에서 파일의 그룹 소유권을 수동으로 수정해야 합니다.

*예 **21.3.** 사용자의 기본 그룹을 변경하는 예:*

***user sarah**가 기본 **group sarah1** 에 속하고 사용자의 기본 그룹을 **sarah2** 로 변경하려면 다음을 사용합니다.*

```
# usermod -g sarah2 sarah
```

검증 단계

- *사용자의 기본 그룹을 변경했는지 확인합니다.*

```
$ groups sarah
```

출력이 표시됩니다.

```
sarah : sarah2
```

***21.4.** 명령줄에서 사용자를 보조 그룹에 추가*

보조 그룹에 사용자를 추가하여 권한을 관리하거나 특정 파일 또는 장치에 대한 액세스를 활성화할 수 있습니다.

사전 요구 사항

- 루트 액세스

절차

- 사용자 보조 그룹에 그룹을 추가하려면 다음을 사용합니다.

```
# usermod --append -G group-name username
```

group-name 을 그룹 이름으로 바꾸고 **username** 을 사용자 이름으로 교체합니다.

예 **21.4.** 보조 그룹에 사용자 추가

사용자 **administrator**를 **system-administrators** 그룹에 추가하려면 다음을 사용합니다.

```
# usermod --append -G system-administrators sysadmin
```

검증 단계

- 새 그룹이 사용자 **administrator**의 보조 그룹에 추가되었는지 확인하려면 다음을 사용합니다.

```
# groups sysadmin
```

출력이 표시됩니다.

```
sysadmin : sysadmin system-administrators
```

21.5. 보조 그룹에서 사용자 제거

보조 그룹에서 기존 사용자를 제거하여 파일 및 장치에 대한 권한을 제한할 수 있습니다.

사전 요구 사항

- *루트 액세스*

절차

- *보조 그룹에서 사용자 제거:*

```
# gpasswd -d user-name group-name
```

***user-name** 을 사용자 이름으로 바꾸고 **group-name** 을 보조 그룹 이름으로 교체합니다.*

> ***예 21.5.** 보조 그룹에서 사용자 제거*
>
> *사용자 **sarah**에 기본 **group sarah2** 가 있고 보조 그룹 **wheel** 및 **developers** 에 속하고 그룹 개발자에서 해당 사용자를 제거하려면 다음을 사용합니다.*
>
> ```
> # gpasswd -d sarah developers
> ```

검증 단계

- *보조 그룹 **developers**에서 사용자 **sarah**가 제거되었는지 확인합니다.*

```
$ groups sarah
```

출력이 표시됩니다.

```
sarah : sarah2 wheel
```

***21.6.** 사용자의 모든 보조 그룹 변경*

사용자가 멤버로 유지하려는 보조 그룹 목록을 덮어쓸 수 있습니다.

사전 요구 사항

- *루트 액세스*

- 보조 그룹이 있어야 합니다.

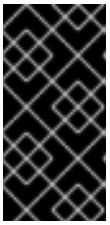
절차

- 사용자의 보조 그룹 목록을 덮어씁니다.

```
# usermod -G group-names username
```

group-names 를 하나 이상의 보조 그룹의 이름으로 바꿉니다. 사용자를 한 번에 여러 보조 그룹에 추가하려면 그룹 이름을 쉼표와 중간 공백을 사용하여 구분합니다. 예: **wheel,developer.**

user-name 을 사용자 이름으로 변경합니다.



중요

사용자가 현재 지정하지 않는 그룹의 멤버인 경우 명령은 그룹에서 사용자를 제거합니다.

예 **21.6.** 사용자의 보조 그룹 목록 변경

user sarah 에 기본 **group sarah2** 가 있고 보조 그룹 **wheel** 에 속하고 사용자가 **3**개의 보조 그룹 **developer,sysadmin** 및 **security** 에 속하게 하려면 다음을 사용합니다.

```
# usermod -G wheel,developer,sysadmin,security sarah
```

검증 단계

- 보조 그룹 목록을 올바르게 설정했는지 확인합니다.

```
# groups sarah
```

출력이 표시됩니다.

```
sarah : sarah2 wheel developer sysadmin security
```

22장. SUDO 액세스 관리

시스템 관리자는 root가 아닌 사용자가 일반적으로 root 사용자를 위해 예약된 관리 명령을 실행할 수 있도록 sudo 액세스 권한을 부여할 수 있습니다. 결과적으로 root가 아닌 사용자는 root 사용자 계정에 로그인하지 않고도 이러한 명령을 실행할 수 있습니다.

22.1. SUDOERS의 사용자 권한 부여

/etc/sudoers 파일은 sudo 명령을 사용하여 어떤 명령을 실행할 수 있는지를 지정합니다. 규칙은 개별 사용자 및 사용자 그룹에 적용할 수 있습니다. 별칭을 사용하여 호스트, 명령 및 사용자 그룹에 대한 규칙 정의를 단순화할 수도 있습니다. 기본 별칭은 /etc/sudoers 파일의 첫 번째 부분에 정의됩니다.

사용자가 sudo 권한을 사용하여 /etc/sudoers 파일에서 허용되지 않는 명령을 실행하려고 하면 시스템은 sudoers에 없는username : user not in the journal log를 기록합니다.

기본 /etc/sudoers 파일은 권한 부여의 정보와 예를 제공합니다. 행 시작에서 # 주석 문자를 제거하여 특정 예제 규칙을 활성화할 수 있습니다. 사용자와 관련된 권한 부여 섹션은 다음 소개로 표시됩니다.

```
## Next comes the main part: which users can run what software on
## which machines  (the sudoers file can be shared between multiple
## systems).
```

다음 형식을 사용하여 새 sudoers 권한을 생성하고 기존 권한 부여를 수정할 수 있습니다.

```
username hostname=path/to/command
```

다음과 같습니다.

- username 은 사용자 또는 그룹의 이름입니다(예: user1 또는 %group1).
- hostname 은 규칙이 적용되는 호스트의 이름입니다.
- path/to/command 는 명령의 전체 절대 경로입니다. 명령 경로 뒤에 해당 옵션을 추가하여 특정 옵션 및 인수가 있는 명령만 실행하도록 사용자를 제한할 수도 있습니다. 옵션을 지정하지 않으면 사용자는 모든 옵션과 함께 명령을 사용할 수 있습니다.

이러한 변수 중 하나를 **ALL** 으로 교체하여 모든 사용자, 호스트 또는 명령에 규칙을 적용할 수 있습니다.

주의

All ALL =(ALL) ALL(ALL) 과 같은 과도하게 허용 규칙을 사용하면 모든 사용자가 모든 호스트의 모든 사용자로 모든 명령을 실행할 수 있습니다. 이로 인해 보안 위험이 발생할 수 있습니다.

연산자를 사용하여 부정적인 인수를 지정할 수 있습니다. 예를 들어 **root** 사용자를 제외한 모든 사용자를 지정하려면 **!root** 를 사용합니다. **allowlists**를 사용하여 특정 사용자, 그룹 및 명령을 허용하는 것은 블록 목록을 사용하여 특정 사용자, 그룹 및 명령을 허용하지 않는 것보다 더 안전합니다. 허용 목록을 사용하면 권한이 없는 새로운 사용자 또는 그룹도 차단됩니다.

주의

사용자가 **alias** 명령을 사용하여 명령 이름을 변경하여 이러한 규칙을 해결할 수 있으므로 명령에 음수 규칙을 사용하지 마십시오.

시스템은 **/etc/sudoers** 파일을 처음부터 끝까지 읽습니다. 따라서 파일에 사용자에 대한 여러 항목이 포함된 경우 항목이 순서대로 적용됩니다. 충돌하는 값이 있는 경우 시스템이 가장 구체적인 일치 항목이 아닌 경우에도 마지막 일치 항목을 사용합니다.

sudoers 에 새 규칙을 추가하는 기본 방법은 **/etc/sudoers** 파일에 규칙을 직접 입력하는 대신 **/etc/sudoers.d/** 디렉토리에 새 파일을 만드는 것입니다. 이는 이 디렉터리의 콘텐츠가 시스템 업데이트 중에 보존되기 때문입니다. 또한 **/etc/sudoers** 파일에서보다 별도의 파일에서 오류를 수정하는 것이 더 쉽습니다. **/etc/sudoers** 파일의 다음 행에 도달하면 **/etc/sudoers.d** 디렉토리의 파일을 읽습니다.

```
#includedir /etc/sudoers.d
```

이 행의 시작 부분에 있는 숫자 기호 **#** 은 구문의 일부이며 행이 주석임을 의미하지는 않습니다. 해당 디렉토리에 있는 파일의 이름은 마침표를 포함하지 않아야 하며 틸드 **~** 로 종료해서는 안 됩니다.

22.2. 사용자에게 SUDO 액세스 권한 부여

시스템 관리자는 **root**가 아닌 사용자가 관리 명령을 실행할 수 있도록 **sudo** 액세스 권한을 부여할 수 있습니다. **sudo** 명령은 **root** 사용자의 암호를 사용하지 않고 사용자에게 관리자 액세스 권한을 제공합니다.

사용자가 관리 명령을 수행해야 하는 경우 **sudo** 를 사용하여 해당 명령 앞에 **.** 그런 다음 명령은 **root** 사용자인 것처럼 실행됩니다.

다음과 같은 제한 사항에 유의하십시오.

- **/etc/sudoers** 구성 파일에 나열된 사용자만 **sudo** 명령을 사용할 수 있습니다.
- 명령은 루트 셸이 아닌 사용자 셸에서 실행됩니다.

사전 요구 사항

- 루트 액세스

절차

1. **root**로 **/etc/sudoers** 파일을 엽니다.

```
# visudo
```

/etc/sudoers 파일은 **sudo** 명령으로 적용되는 정책을 정의합니다.

2. **/etc/sudoers** 파일에서 관리 **wheel** 그룹의 사용자에게 **sudo** 액세스 권한을 부여하는 행을 찾습니다.

```
## Allows people in group wheel to run all commands
%wheel        ALL=(ALL)       ALL
```

3. *%wheel 로 시작하는 줄에 # 주석 문자가 없는지 확인합니다.*

4. *변경 사항을 저장하고 편집기를 종료합니다.*

5. *관리 wheel 그룹에 sudo 액세스 권한을 부여하려는 사용자를 추가합니다.*

```
# usermod --append -G wheel username
```

username 을 사용자 이름으로 바꿉니다.

검증 단계

- *사용자가 관리 wheel 그룹에 추가되었는지 확인합니다.*

```
# id username
uid=5000(username) gid=5000(_username) groups=5000(username),10(wheel)
```

22.3. 권한이 없는 사용자가 특정 명령을 실행하도록 활성화

권한이 없는 사용자가 특정 워크스테이션에서 특정 명령을 실행하도록 허용하는 정책을 구성할 수 있습니다. 이 정책을 구성하려면 sudoers.d 디렉터리에 파일을 만들고 편집해야 합니다.

사전 요구 사항

- *루트 액세스*

절차

1. *root로 /etc/ 아래에 새 sudoers.d 디렉토리를 만듭니다.*

```
# mkdir -p /etc/sudoers.d/
```

2. */etc/sudoers.d 디렉토리에 새 파일을 만듭니다.*

```
# visudo -f /etc/sudoers.d/file-name
```

file-name 을 생성할 파일의 이름으로 바꿉니다. 파일이 자동으로 열립니다.

3. 새로 생성된 파일에 다음 행을 추가합니다.

```
username hostname = /path/to/the/command
```

username 을 사용자 이름으로 바꿉니다. **hostname** 을 호스트 이름으로 교체합니다. **/path/to/the/command** 를 명령의 절대 경로(예: **/usr/bin/dnf**)로 바꿉니다.

4. 변경 사항을 저장하고 편집기를 종료합니다.

예 22.1. 권한이 없는 사용자가 **dnf**를 사용하여 프로그램을 설치할 수 있음

사용자 **sarah** 가 **sudo** 권한이 있는 **dnf** 유틸리티를 사용하여 **localhost.localdomain** 워크스테이션에 프로그램을 설치할 수 있도록 하려면 다음을 사용합니다.

1. **root**로 **/etc/** 아래에 새 **sudoers.d** 디렉토리를 만듭니다.

```
# mkdir -p /etc/sudoers.d/
```

2. **/etc/sudoers.d** 디렉토리에 새 파일을 만듭니다.

```
# visudo -f /etc/sudoers.d/sarah
```

파일이 자동으로 열립니다.

3. **/etc/sudoers.d/sarah** 파일에 다음 행을 추가합니다.

```
sarah localhost.localdomain = /usr/bin/dnf
```

두 명령 경로가 , 쉼표 뒤에 공백 으로 구분되어 있는지 확인합니다.

4. 선택 사항: 사용자 **sarah** 가 **sudo** 권한을 사용하려고 할 때마다 이메일 알림을 받으려면 파일에 다음 행을 추가합니다.

```
Defaults    mail_always
Defaults    mailto="email@domain.com"
```

5. **user sarah** 가 **sudo** 권한으로 **dnf** 명령을 실행할 수 있는지 확인하려면 계정을 전환합니다.

```
# su sarah -
```

6. **sudo dnf** 명령을 입력합니다.

```
$ sudo dnf
[sudo] password for sarah:
```

사용자 **sarah** 의 **sudo** 암호를 입력합니다.

7. 시스템에 **dnf** 명령 및 옵션 목록이 표시됩니다.

```
...
usage: dnf [options] COMMAND
...
```

sarah가 **sudoers** 파일에 없는 경우. 이 문제는 보고됩니다. 메시지는 구성이 올바르게 완료되지 않았습니다. 이 절차를 **root** 로 실행하고 단계를 철저하게 수행했는지 확인하십시오.

22.4. 추가 리소스

- **sudo(8)** 도움말 페이지
- **visudo(8)** 도움말 페이지

23장. 루트 암호 변경 및 재설정

기존 **root** 암호가 더 이상 적합하지 않거나 잊혀진 경우 **root** 사용자와 비 루트 사용자 모두 변경 또는 재설정할 수 있습니다.

23.1. ROOT 사용자로 ROOT 암호 변경

이 섹션에서는 **passwd** 명령을 사용하여 **root** 사용자로 암호를 변경하는 방법을 설명합니다.

사전 요구 사항

- 루트 액세스

절차

- 루트 암호를 변경하려면 다음을 사용합니다.

```
# passwd
```

암호를 변경하기 전에 현재 암호를 입력하라는 메시지가 표시됩니다.

23.2. 루트가 아닌 사용자로 잊혀진 루트 암호 변경 또는 재설정

이 섹션에서는 **passwd** 명령을 사용하여 잊혀진 루트 암호를 변경하거나 재설정하는 방법을 루트 가 아닌 사용자로 설정하는 방법에 대해 설명합니다.

사전 요구 사항

- 루트가 아닌 사용자로 로그인할 수 있습니다.
- 당신은 관리 위치 그룹의 멤버입니다.

절차

- **wheel** 그룹에 속하는 루트가 아닌 사용자로 **root** 암호를 변경하거나 재설정하려면 다음을 사용합니다.

```
$ sudo passwd root
```

root 암호를 변경하기 전에 현재 **root** 가 아닌 암호를 입력하라는 메시지가 표시됩니다.

23.3. 부팅 시 루트 암호 재설정

루트가 아닌 사용자로 로그인할 수 없거나 관리 **wheel** 그룹에 속하지 않는 경우 특수 **chroot would** 환경으로 전환하여 부팅 시 루트 암호를 재설정할 수 있습니다.

절차

1. 시스템을 재부팅하고 **GRUB 2** 부팅 화면에서 **e** 키를 눌러 부팅 프로세스를 중단합니다.

   커널 부팅 매개변수가 나타납니다.

```
load_video
set gfx_payload=keep
insmod gzio
linux ($root)/vmlinuz-4.18.0-80.e18.x86_64 root=/dev/mapper/rhel-root ro crash\
kernel=auto resume=/dev/mapper/rhel-swap rd.lvm.lv/swap rhgb quiet
initrd ($root)/initramfs-4.18.0-80.e18.x86_64.img $tuned_initrd
```

2. **linux** 로 시작하는 행 끝으로 이동하십시오.

```
linux ($root)/vmlinuz-4.18.0-80.e18.x86_64 root=/dev/mapper/rhel-root ro crash\
kernel=auto resume=/dev/mapper/rhel-swap rd.lvm.lv/swap rhgb quiet
```

   Ctrl+e 를 눌러 행 끝으로 건너뜁니다.

3. **linux** 로 시작하는 행 끝에 **rd.break** 를 추가합니다.

```
linux ($root)/vmlinuz-4.18.0-80.e18.x86_64 root=/dev/mapper/rhel-root ro crash\
kernel=auto resume=/dev/mapper/rhel-swap rd.lvm.lv/swap rhgb quiet rd.break
```

4. **Ctrl+x** 를 눌러 변경된 매개 변수를 사용하여 시스템을 시작합니다.

switch_root 프롬프트가 나타납니다.

5. 파일 시스템을 쓰기 가능으로 다시 마운트합니다.

```
mount -o remount,rw /sysroot
```

파일 시스템은 **/sysroot** 디렉토리에서 읽기 전용으로 마운트됩니다. 파일 시스템을 쓰기 가능으로 다시 마운트하면 암호를 변경할 수 있습니다.

6. **chroot** 환경을 입력합니다.

```
chroot /sysroot
```

sh-4.4# 프롬프트가 표시됩니다.

7. 루트 암호를 재설정합니다.

```
passwd
```

명령줄에 표시된 지침에 따라 **root** 암호 변경을 완료합니다.

8. 다음 시스템 부팅 시 **SELinux** 재지정 프로세스를 활성화합니다.

```
touch /.autorelabel
```

9. **chroot** 환경을 종료합니다.

```
exit
```

10. **switch_root** 프롬프트를 종료합니다.

```
exit
```

11.

SELinux 레이블 지정 프로세스가 완료될 때까지 기다립니다. 큰 디스크의 레이블을 다시 지정하는 데 시간이 오래 걸릴 수 있습니다. 프로세스가 완료되면 시스템이 자동으로 재부팅됩니다.

검증 단계

1. *루트 암호가 변경되었는지 확인하려면 일반 사용자로 로그인하여 터미널을 엽니다.*

2. *대화형 쉘을 root로 실행합니다.*

```
$ su
```

3. *새 루트 암호를 입력합니다.*

4. *현재 유효한 사용자 ID와 관련된 사용자 이름을 인쇄합니다.*

```
whoami
```

출력이 반환됩니다.

```
root
```

24장. 파일 권한 관리

파일 권한은 파일 및 디렉터리의 내용을 확인, 수정, 액세스 및 실행하는 사용자 및 그룹 계정의 기능을 제어합니다.

모든 파일 또는 디렉터리에는 세 가지 수준의 소유권이 있습니다.

- 사용자 소유자(u).
- 그룹 소유자(g).
- 기타(o).

각 수준의 소유권에는 다음 권한이 할당될 수 있습니다.

- 읽기(r).
- 쓰기(W).
- 실행(x).

파일에 대한 실행 권한을 사용하면 해당 파일을 실행할 수 있습니다. 디렉터리에 대한 실행 권한을 사용하면 디렉터리의 콘텐츠에 액세스할 수 있지만 실행할 수는 없습니다.

새 파일 또는 디렉터리가 생성되면 기본 권한 집합이 자동으로 할당됩니다. 파일 또는 디렉터리에 대한 기본 권한은 다음 두 가지 요인을 기반으로 합니다.

- 기본 권한.
- 사용자 파일 생성 모드 마스크(mask)입니다.

24.1. 기본 파일 권한

새 파일이나 디렉터리가 생성될 때마다 기본 권한이 자동으로 할당됩니다. 파일 또는 디렉터리에 대한 기본 권한을 기호 또는 8진수 값으로 표시할 수 있습니다.

권한	심볼릭 값	8진수 값
권한 없음	---	0
execute	--x	1
write	-w-	2
쓰기 및 실행	-wx	3
읽기	r--	4
읽기 및 실행	r-x	5
읽기 및 쓰기	rw-	6
읽기, 쓰기, 실행	rwx	7

디렉터리에 대한 기본 권한은 **777 (drwxrwxrwx)**이며 모든 사용자에게 읽기, 쓰기, 실행 권한을 부여합니다. 즉, 디렉터리 소유자, 그룹 및 기타 사용자가 디렉터리의 콘텐츠를 나열하고, 디렉터리 내의 항목을 만들고, 삭제하고, 편집하고, 편집할 수 있습니다.

디렉토리에 있는 개별 파일에는 디렉터리에 대한 무제한 액세스 권한이 있더라도 편집할 수 있는 자체 권한이 있을 수 있습니다.

파일에 대한 기본 권한은 **666 (-rw-rw-rw-)**이며 모든 사용자에게 읽기 및 쓰기 권한을 부여합니다. 즉, 파일 소유자, 그룹 및 다른 사용자가 파일을 읽고 편집할 수 있습니다.

> **예 24.1.** 파일에 대한 권한
>
> 파일에 다음 권한이 있는 경우:
>
> ```
> $ ls -l
> -rwxrw----. 1 sysadmins sysadmins 2 Mar 2 08:43 file
> ```

- *- 파일이 있음을 나타냅니다.*
- ***rwx** 는 파일 소유자가 파일을 읽고, 쓰고, 실행할 수 있는 권한이 있음을 나타냅니다.*
- ***RW-** 는 그룹에 읽기 및 쓰기 권한이 있지만 파일을 실행하지는 않음을 나타냅니다.*
- ***---** 는 다른 사용자가 파일을 읽고, 쓰고, 실행할 수 있는 권한이 없음을 나타냅니다.*
- *. 는 **SELinux** 보안 컨텍스트가 파일에 설정되어 있음을 나타냅니다.*

*예 **24.2.** 디렉터리에 대한 권한*

디렉터리에 다음 권한이 있는 경우:

```
$ ls -dl directory
drwxr-----. 1 sysadmins sysadmins 2 Mar 2 08:43 directory
```

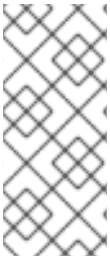
- ***D** 는 디렉터리임을 나타냅니다.*
- ***rwx** 는 디렉터리 소유자가 디렉터리의 콘텐츠를 읽고, 쓰고, 액세스할 수 있는 권한이 있음을 나타냅니다.*

  디렉터리 소유자는 디렉터리 내의 항목(파일, 하위 디렉터리)을 나열하고 해당 항목의 콘텐츠에 액세스한 다음 수정할 수 있습니다.
- ***R--** 는 그룹에 읽기 권한이 있지만 디렉터리의 내용을 쓰거나 액세스하지 않음을 나타냅니다.*

  디렉터리가 속한 그룹의 구성원으로 디렉터리 내의 항목을 나열할 수 있습니다. 디렉터리 내의 항목에 액세스하거나 수정할 수 없습니다.
- ***---** 는 다른 사용자가 디렉터리의 콘텐츠를 읽고, 쓰거나, 액세스할 수 있는 권한이 없음을 나타냅니다.*

사용자 소유자가 아니거나 디렉토리의 그룹 소유자로서는 디렉터리 내의 항목을 나열하거나, 해당 항목에 대한 정보에 액세스하거나, 수정할 수 없습니다.

- *. 는 **SELinux** 보안 컨텍스트가 디렉터리에 설정되어 있음을 나타냅니다.*



참고

*파일 또는 디렉터리에 자동으로 할당된 기본 권한은 파일 또는 디렉터리가 끝나는 기본 권한이 아닙니다. 파일 또는 디렉토리를 생성하면 기본 권한이 **permissions**에 의해 변경됩니다. 기본 권한과 **mTLS**의 조합은 파일 및 디렉터리에 대한 기본 권한을 생성합니다.*

24.2. 사용자 파일 생성 모드 마스크

*사용자 파일 생성 모드 마스크(**balancer**)는 새로 생성된 파일 및 디렉터리에 대해 파일 권한이 설정되는 방법을 제어하는 변수입니다. **imagestreamtag** 는 **Linux** 시스템의 전체 보안을 높이기 위해 기본 권한 값에서 권한을 자동으로 제거합니다. **jaeger** 는 심볼릭 또는 **8**진수 값으로 표현할 수 있습니다.*

권한	심볼릭 값	8진수 값
읽기, 쓰기 및 실행	rwx	0
읽기 및 쓰기	rw-	1
읽기 및 실행	r-x	2
읽기	r--	3
쓰기 및 실행	-wx	4
write	-w-	5
execute	--x	6
권한 없음	---	7

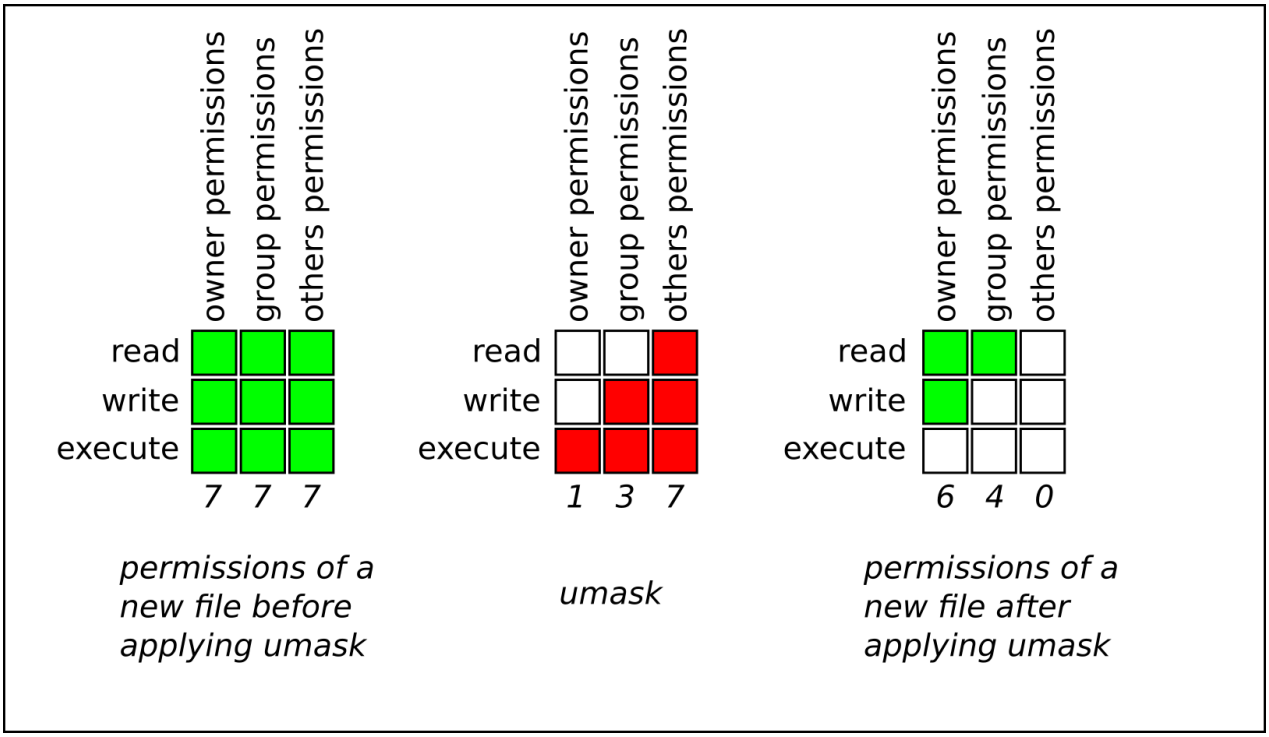
*표준 사용자의 기본 **mTLS**는 **0002** 입니다. **root** 사용자의 기본 **mTLS**는 **0022** 입니다.*

***messages**의 첫 번째 숫자는 특수 권한(**sticky bit,)**을 나타냅니다. **PATH**의 마지막 세 자리는 사용자*

소유자(u), 그룹 소유자(g) 및 기타(o)에서 각각 제거된 권한을 나타냅니다.

예 24.3. 파일을 생성할 때 mTLS 적용

다음 예제에서는 기본 권한이 777 인 파일에 8진수 값이 0137 인 umask 를 적용하여 기본 권한이 640 인 파일을 생성하는 방법을 보여줍니다.



24.3. 기본 파일 권한

새로 생성된 모든 파일 및 디렉터리에 대해 기본 권한이 자동으로 설정됩니다. 기본 권한의 값은 기본 권한에 umask 를 적용하여 결정됩니다.

예 24.4. 표준 사용자가 생성한 디렉터리에 대한 기본 권한

표준 사용자가 새 디렉토리를 생성하면 mTLS가 002 (rwxr-x)로 설정되고 디렉토리의 기본 권한은 777 (rwxrwx)으로 설정됩니다. 기본 권한을 775 (drwxr-x)로 가져옵니다.

	심볼릭 값	8진수 값
기본 권한	rwxrwxrwx	777
jaeger	rwxrwxr-x	002

기본 권한	rwxrwxr-x	775

즉, 디렉터리 소유자와 그룹은 디렉터리의 콘텐츠를 나열하고, 디렉터리 내의 항목을 생성, 삭제, 편집할 수 있으며, 이 디렉터리에 내 항목을 내림차순할 수 있습니다. 다른 사용자는 디렉터리의 콘텐츠만 나열하고 디렉토리 내림차순을 나열할 수 있습니다.

예 24.5. 표준 사용자가 생성한 파일에 대한 기본 권한

*표준 사용자가 새 파일 을 생성할 때 **mTLS**는 **002 (rwxr-x)**로 설정되고 파일의 기본 권한이 **666 (rw-rw-rw-)**으로 설정됩니다. 그러면 기본 권한이 **664 (-rw-rw-r--)**가 됩니다.*

	심볼릭 값	8진수 값
기본 권한	rw-rw-rw-	666
jaeger	rwxrwxr-x	002
기본 권한	rw-rw-r--	664

*즉, **file owner**와 **group**은 파일을 읽고 편집할 수 있지만 다른 사용자는 파일만 읽을 수 있습니다.*

*예 24.6. **root** 사용자가 생성한 디렉터리에 대한 기본 권한*

*루트 사용자가 새 디렉토리를 생성할 때 루트 사용자가 **022 (rwxr-xr-x)**로 설정되고, 디렉터리에 대한 기본 권한은 **777 (rwxrwx)**으로 설정됩니다. 그러면 기본 권한이 **755 (rwxr-xr-x)**가 됩니다.*

	심볼릭 값	8진수 값
기본 권한	rwxrwxrwx	777
jaeger	rwxr-xr-x	022
기본 권한	rwxr-xr-x	755

즉, 디렉터리 소유자는 디렉터리의 콘텐츠를 나열하고, 디렉터리 내의 항목을 생성, 삭제, 편집할 수 있으며, 디렉터리의 콘텐츠를 내림차순할 수 있습니다. 그룹 및 기타 그룹은 디렉터리의 콘텐츠만 나열하고 그 내용을 추측할 수 있습니다.

예 24.7. root 사용자가 생성한 파일에 대한 기본 권한

루트 사용자가 새 파일 을 생성할 때 루트 사용자가 022 (rwxr-xr-x)로 설정되고 파일의 기본 권한이 666 (rw-rw-rw-)로 설정됩니다. 그러면 기본 권한이 644 (-rw-r-do)로 표시됩니다.r--).

	심볼릭 값	8진수 값
기본 권한	rw-rw-rw-	666
jaeger	rwxr-xr-x	022
기본 권한	rw-r—r--	644

즉, 파일 소유자는 파일을 읽고 편집할 수 있으며, 그룹 및 다른 사용자는 파일을 읽을 수 있습니다.

참고

보안상의 이유로 permissions가 000 (rwxrwx)으로 설정되어 있어도 일반 파일은 기본적으로 실행 권한을 가질 수 없습니다. 그러나 실행 권한을 사용하여 디렉터리를 만들 수 있습니다.

24.4. 심볼릭 값을 사용하여 파일 권한 변경

tekton 유틸리티를 심볼릭 값(조합 문자 및 기호)과 함께 사용하여 파일 또는 디렉토리에 대한 파일 권한을 변경할 수 있습니다.

다음 권한을 할당할 수 있습니다.

- *읽기(r)*
- *쓰기(W)*
- *실행 (x)*

권한은 다음 수준의 소유권 에 할당할 수 있습니다.

- 사용자 소유자 (u)
- 그룹 소유자(g)
- 기타 (O)
- 모두 (a)

사용 권한 추가 또는 제거 하려면 다음 표시를 사용 합니다.To add or remove permissions you can use the following signs:

- + 기존 권한 상단에 대한 권한을 추가하려면 다음을 수행합니다.
- 기존 권한에서 권한을 제거하려면To remove the permissions from the existing permission
- 기존 권한을 제거하고 새 권한을 명시적으로 정의하려면 =

절차

- 파일 또는 디렉터리에 대한 권한을 변경하려면 다음을 사용합니다.

```
$ chmod <level><operation><permission> file-name
```

<level> 을 권한을 설정하려는 소유권 수준으로 바꿉니다. <operation> 를 부호 중 하나로 바꿉니다. <permission> 를 할당할 권한으로 바꿉니다. file-name 을 파일 또는 디렉터리의 이름으로 바꿉니다. 예를 들어 모든 사용자에게 읽기, 쓰기, 실행(rwx) my-script.sh 를 부여하려면 etcdctl a=rwx my-script.sh 명령을 사용합니다.

자세한 내용은 기본 파일 권한을 참조하십시오.

검증 단계

- *특정 파일에 대한 권한을 보려면 다음을 사용합니다.*

```
$ ls -l file-name
```

  ***file-name** 을 파일 이름으로 바꿉니다.*

- *특정 디렉터리에 대한 권한을 보려면 다음을 사용합니다.*

```
$ ls -dl directory-name
```

  ***directory-name** 을 디렉터리 이름으로 바꿉니다.*

- *특정 디렉터리 내의 모든 파일에 대한 권한을 보려면 다음을 사용합니다.*

```
$ ls -l directory-name
```

  ***directory-name** 을 디렉터리 이름으로 바꿉니다.*

*예 **24.8.** 파일 및 디렉터리에 대한 권한 변경*

- ***my-file.txt** 의 파일 권한을 **-rw-rw-r---** 에서 **-rw-----** 으로 변경하려면 다음을 사용합니다.*

  1. ***my-file.txt** 에 대한 현재 권한을 표시합니다.*

```
$ ls -l my-file.txt
-rw-rw-r--. 1 username username 0 Feb 24 17:56 my-file.txt
```

  2. *그룹 소유자(**g**) 및 기타(**o**)에서 파일을 읽고 쓰고 실행할 수 있는 권한을 제거합니다.*

```
$ chmod go= my-file.txt
```

     *등호(**=**) 후에 지정되지 않은 권한은 자동으로 금지됩니다.*

3. *my-file.txt 에 대한 권한이 올바르게 설정되었는지 확인합니다.*

```
$ ls -l my-file.txt
-rw-------. 1 username username 0 Feb 24 17:56 my-file.txt
```

- *my-directory 의 파일 권한을 drwxrwx--- 에서 drwxrwxr-x 로 변경하려면 다음을 사용합니다.*

1. *my-directory 에 대한 현재 권한을 표시합니다.*

```
$ ls -dl my-directory
drwxrwx---. 2 username username 4096 Feb 24 18:12 my-directory
```

2. *모든 사용자(a)에 대해 읽기 및 실행(r-x) 액세스를 추가합니다.*

```
$ chmod o+rx my-directory
```

3. *my-directory 및 해당 콘텐츠에 대한 권한이 올바르게 설정되었는지 확인합니다.*

```
$ ls -dl my-directory
drwxrwxr-x. 2 username username 4096 Feb 24 18:12 my-directory
```

24.5. 8진수 값을 사용하여 파일 권한 변경

8진수 값(numbers)과 함께 tekton 유틸리티를 사용하여 파일 또는 디렉터리에 대한 파일 권한을 변경할 수 있습니다.

절차

- *기존 파일 또는 디렉터리의 파일 권한을 변경하려면 다음을 사용합니다.*

```
$ chmod octal_value file-name
```

file-name 을 파일 또는 디렉터리의 이름으로 바꿉니다. 8진수_value 를 8진수 값으로 바꿉니다. 자세한 내용은 기본 파일 권한을 참조하십시오.

25장. RECEIVER 관리

umask 유틸리티를 사용하여 umask 의 현재 또는 기본값을 표시, 설정 또는 변경할 수 있습니다.

25.1. MTLS의 현재 값 표시

dependencies 유틸리티를 사용하여 현재 데이터 값을 심볼릭 또는 8진수 모드로 표시할 수 있습니다.

절차

- symbolic mode로 topology의 현재 값을 표시하려면 다음을 사용합니다.

```
$ umask -S
```

- 8진수 모드에서 topology의 현재 값을 표시하려면 다음을 사용합니다.

```
$ umask
```

참고

8진수 모드로 umask 를 표시할 때 4자리 숫자(0002 또는 0022)로 표시될 수 있습니다. messages의 첫 번째 숫자는 특수 비트 (sticky bit, SGID 비트 또는 SUID 비트)를 나타냅니다. 첫 번째 숫자가 0 으로 설정되면 특수 비트가 설정되지 않습니다.

25.2. 기본 BASH CREDENTIALS 표시

bash,ksh,zsh 및 tcsh와 같이 사용할 수 있는 쉘은 여러 가지가 있습니다. 이러한 쉘은 로그인 또는 비로그인 쉘로 작동할 수 있습니다. 네이티브 또는 GUI 터미널을 열어 로그인 쉘을 호출할 수 있습니다.

로그인 또는 비로그인 쉘에서 명령을 실행 중인지 확인하려면 echo $0 명령을 사용합니다.

예 25.1. 로그인 또는 비로그인 bash 쉘에서 작동하는지 확인

- echo $0 명령의 출력이 bash 를 반환하면 비로그인 쉘에서 명령을 실행하고 있습니다.

```
$ echo $0
bash
```

로그인이 아닌 쉘의 기본 **umask** 는 **/etc/bashrc** 구성 파일에 설정됩니다.

- **echo $0** 명령의 출력이 **-bash** 를 반환하는 경우 로그인 쉘에서 명령을 실행합니다.

```
# echo $0
-bash
```

로그인 쉘의 기본 **umask** 는 **/etc/login.defs** 구성 파일에 설정되어 있습니다.

절차

- 로그인이 아닌 쉘에 대한 기본 **bash umask** 를 표시하려면 다음을 사용합니다.

```
$ grep umask /etc/bashrc
```

출력이 반환됩니다.

```
# By default, we want umask to get set. This sets it for non-login shell.
      umask 002
      umask 022
```

- 로그인 쉘에 대한 기본 **bash umask** 를 표시하려면 다음을 사용합니다.

```
grep "UMASK" /etc/login.defs
```

출력이 반환됩니다.

```
# UMASK is also used by useradd(8) and newusers(8) to set the mode for new
UMASK        022
# If HOME_MODE is not set, the value of UMASK is used to create the mode.
```

25.3. 심볼릭 값을 사용하여 MTLS 설정

symlink 유틸리티의 심볼릭 값(스크립 문자 및 기호)을 사용하여 현재 셸 세션에 대한 umask 를 설정할 수 있습니다.

다음 권한을 할당할 수 있습니다.

- *읽기(r)*
- *쓰기(W)*
- *실행 (x)*

권한은 다음 수준의 소유권 에 할당할 수 있습니다.

- *사용자 소유자 (u)*
- *그룹 소유자(g)*
- *기타 (O)*
- *모두 (a)*

사용 권한 추가 또는 제거 하려면 다음 표시를 사용 합니다.To add or remove permissions you can use the following signs:

- *+ 기존 권한 상단에 대한 권한을 추가하려면 다음을 수행합니다.*
- *기존 권한에서 권한을 제거하려면To remove the permissions from the existing permission*
- *기존 권한을 제거하고 새 권한을 명시적으로 정의하려면 =*

참고

등호(=) 이후에 지정되지 않은 모든 권한은 자동으로 금지됩니다.

절차

- 현재 쉘 세션에 대해 **mTLS**를 설정하려면 다음을 사용합니다.

```
$ umask -S <level><operation><permission>
```

<level> 을 에 대해 설정하려는 소유권 레벨로 바꿉니다. **<operation>** 를 부호 중 하나로 바꿉니다. **<permission>** 를 할당할 권한으로 바꿉니다. 예를 들어, **permissions**를 **u=rwx,g=rwx,o=rwx** 으로 설정하려면 **umask -S a=rwx** 를 사용합니다.

자세한 내용은 사용자 파일 생성 모드를 참조하십시오.

참고

jaeger 는 현재 쉘 세션에만 유효합니다.

25.4. 8진수 값을 사용하여 RECEIVER 설정

8진수 값**(number)**과 함께 **wizard** 유틸리티를 사용하여 현재 쉘 세션에 대한 **umask** 를 설정할 수 있습니다.

절차

- 현재 쉘 세션에 대해 **mTLS**를 설정하려면 다음을 사용합니다.

```
$ umask octal_value
```

8진수**_value** 를 **8**진수 값으로 바꿉니다. 자세한 내용은 **User file-creation mode mask** 를 참조하십시오.

참고

jaeger 는 현재 쉘 세션에만 유효합니다.

25.5. 로그인이 아닌 쉘의 기본 MTLS 변경

/etc/bashrc 파일을 수정하여 표준 사용자의 기본 **bash permissions**를 변경할 수 있습니다.

사전 요구 사항

- 루트 액세스

절차

1. 루트 로서 편집기에서 **/etc/bashrc** 파일을 엽니다.
2. 다음 섹션을 수정하여 새로운 기본 **bash umask** 를 설정합니다.

```
if [ $UID -gt 199 ] && [ “id -gn” = “id -un” ]; then
   umask 002
else
   umask 022
fi
```

errata (002)의 기본 **8**진수 값을 다른 **8**진수 값으로 바꿉니다. 자세한 내용은 **User file-creation mode mask** 를 참조하십시오.

3. 변경 사항을 저장하고 편집기를 종료합니다.

25.6. 로그인 쉘의 기본 MTLS 변경

/etc/login.defs 파일을 수정하여 **root** 사용자의 기본 **bash umask** 를 변경할 수 있습니다.

사전 요구 사항

- 루트 액세스

절차

1. *루트 권한으로 편집기에서 **/etc/login.defs** 파일을 엽니다.*

2. *다음 섹션을 수정하여 새로운 기본 **bash umask** 를 설정합니다.*

```
# Default initial "umask" value used by login(1) on non-PAM enabled systems.
# Default "umask" value for pam_umask(8) on PAM enabled systems.
# UMASK is also used by useradd(8) and newusers(8) to set the mode for new
# home directories if HOME_MODE is not set.
# 022 is the default value, but 027, or even 077, could be considered
# for increased privacy. There is no One True Answer here: each sysadmin
# must make up their mind.

UMASK           022
```

   ***umask (022)**의 기본 **8**진수 값을 다른 **8**진수 값으로 교체합니다. 자세한 내용은 **User file-creation mode mask** 를 참조하십시오.*

3. *변경 사항을 저장하고 편집기를 종료합니다.*

25.7. 특정 사용자의 기본 MTLS 변경

*해당 사용자의 **. bashrc** 를 수정하여 특정 사용자의 기본 **mTLS**를 변경할 수 있습니다.*

절차

- *특정 사용자의 **.bashrc** 파일에") 의 **8**진수 값을 지정하는 행을 추가합니다.*

```
$ echo 'umask octal_value' >> /home/username/.bashrc
```

   ***8**진수_**value** 를 **8**진수 값으로 바꾸고 **username** 을 사용자 이름으로 교체합니다. 자세한 내용은 **User file-creation mode mask** 를 참조하십시오.*

25.8. 새로 생성된 홈 디렉터리에 대한 기본 권한 설정

*/**etc/login.defs** 파일을 수정하여 새로 생성된 사용자의 홈 디렉토리에 대한 권한 모드를 변경할 수 있습니다.*

절차

1. *루트 권한으로 편집기에서 **/etc/login.defs** 파일을 엽니다.*

2. *다음 섹션을 수정하여 새로운 기본 **HOME_MODE** 를 설정합니다.*

```
# HOME_MODE is used by useradd(8) and newusers(8) to set the mode for new
# home directories.
# If HOME_MODE is not set, the value of UMASK is used to create the mode.
HOME_MODE       0700
```

*기본 **8**진수 값**(0700)**을 다른 **8**진수 값으로 교체합니다. 선택한 모드는 홈 디렉토리에 대한 권한을 생성하는 데 사용됩니다.*

3. ***HOME_MODE** 가 설정되어 있으면 변경 사항을 저장하고 편집기를 종료합니다.*

4. ***HOME_MODE** 가 설정되지 않은 경우 **UMASK** 를 수정하여 새로 생성된 홈 디렉터리의 모드를 설정합니다.*

```
# Default initial "umask" value used by login(1) on non-PAM enabled systems.
# Default "umask" value for pam_umask(8) on PAM enabled systems.
# UMASK is also used by useradd(8) and newusers(8) to set the mode for new
# home directories if HOME_MODE is not set.
# 022 is the default value, but 027, or even 077, could be considered
# for increased privacy. There is no One True Answer here: each sysadmin
# must make up their mind.

UMASK           022
```

*기본 **8**진수 값**(022)**을 다른 **8**진수 값으로 바꿉니다. 자세한 내용은 **User file-creation mode mask** 를 참조하십시오.*

5. *변경 사항을 저장하고 편집기를 종료합니다.*

26장. 액세스 제어 목록 관리

각 파일과 디렉토리는 한 번에 하나의 사용자 소유자와 그룹 소유자만 가질 수 있습니다. 다른 파일 및 디렉터리를 비공개로 유지하면서 다른 사용자 또는 그룹에 속하는 특정 파일 또는 디렉터리에 액세스할 수 있는 권한을 사용자에게 부여하려면 **Linux ACL(액세스 제어 목록)**을 사용할 수 있습니다.

26.1. 현재 액세스 제어 목록 표시

getfacl 유틸리티를 사용하여 현재 **ACL**을 표시할 수 있습니다.

절차

- 특정 파일 또는 디렉터리에 대한 현재 **ACL**을 표시하려면 다음을 사용합니다.

```
$ getfacl file-name
```

file-name 을 파일 또는 디렉터리의 이름으로 바꿉니다.

26.2. 액세스 제어 목록 설정

setfacl 유틸리티를 사용하여 파일 또는 디렉터리에 대한 **ACL**을 설정할 수 있습니다.

사전 요구 사항

- 루트 액세스입니다.

절차

- 파일 또는 디렉터리에 대해 **ACL**을 설정하려면 다음을 사용합니다.

```
# setfacl -m u:username:symbolic_value file-name
```

username 을 사용자 이름으로, **symbolic_value** 를 심볼릭 값으로 바꾸고 **file-name** 을 파일 또는 디렉터리 이름으로 바꿉니다. 자세한 내용은 **setfacl man** 페이지를 참조하십시오.

예 26.1. 그룹 프로젝트에 대한 권한 수정

*다음 예제에서는 이 파일이 **root** 그룹에 속하는 **root** 사용자가 소유한 **group-project** 파일에 대한 권한을 수정하는 방법을 설명합니다.*

- *모든 사람이 실행할 수 없습니다.*
- *사용자 및 **re w**에는 **rw-** 권한이 있습니다.*
- *사용자 **susan** 에는 **---** 권한이 있습니다.*
- *다른 사용자에게는 **r--** 권한이 있습니다.*

절차

```
# setfacl -m u:andrew:rw- group-project
# setfacl -m u:susan:--- group-project
```

검증 단계

- *사용자 및**rew**에 **rw-** 권한이 있는지 확인하려면 사용자 **susan** 에 **---** 권한이 있으며 기타 사용자에게는 **r--** 권한이 있습니다.*

```
$ getfacl group-project
```

출력이 반환됩니다.

```
# file: group-project
# owner: root
# group: root
user:andrew:rw-
user:susan:---
group::r--
mask::rw-
other::r--
```

27장. CHRONY 모음을 사용하여 NTP 구성

정확한 시간 유지는 IT에서 여러 가지 이유로 중요합니다. 예를 들어 네트워킹에서는 패킷 및 로그의 정확한 타임스탬프가 필요합니다. Linux 시스템에서 NTP 프로토콜은 사용자 공간으로 실행되는 데몬에 의해 구현됩니다.

사용자 공간 데몬은 커널에서 실행되는 시스템 클럭을 업데이트합니다. 시스템 클럭은 다양한 클럭 소스를 사용하여 시간을 유지할 수 있습니다. 일반적으로TSC(Time Stamp TSC)가 사용됩니다. TSC는 마지막 재설정 이후 사이클 수를 계산하는 CPU 레지스터입니다. 매우 빠르고 높은 해상도가 있으며 중단이 없습니다.

Red Hat Enterprise Linux 8부터 NTP 프로토콜은 chronyd 데몬에서 구현되며 chrony 패키지의 리포지토리에서 사용할 수 있습니다.

다음 섹션에서는 chrony 제품군을 사용하여 NTP를 설정하는 방법을 설명합니다.

27.1. CHRONY 제품군 소개

Chrony 는 NTP(Network Time Protocol) 구현입니다. chrony 를 사용할 수 있습니다:

- 시스템 클럭을 NTP 서버와 동기화하려면
- 시스템 클럭을 참조 클럭과 동기화하기 위해, 예를 들면 GPS 수신기
- 시스템 클럭을 수동 시간 입력과 동기화하려면
- NTPv4(RFC 5905) 서버 또는 피어로서 네트워크의 다른 컴퓨터에 시간 서비스를 제공합니다.

chrony 는 간헐적인 네트워크 연결, 과도한 정체된 네트워크, 온도 변경(또는 기존 컴퓨터 클럭이 온도에 민감함) 및 가상 머신에서 실행되지 않는 시스템을 포함한 다양한 조건에서 잘 작동합니다.

인터넷을 통해 동기화된 두 시스템 간의 일반적인 정확도는 몇 밀리초 내에 있으며 10초 이내에 LAN의 경우 입니다. 하드웨어 타임스탬프 또는 하드웨어 참조 클럭은 마이크로초 수준에 동기화된 두 시스템 간

의 정확도를 향상시킬 수 있습니다.

chrony 는 사용자 공간으로 실행되는 데몬인 chronyd, chronyc, chronyd 의 성능을 모니터링하고 실행 시 다양한 운영 매개 변수를 변경하는 데 사용할 수 있는 명령행 프로그램으로 구성됩니다.

chrony 데몬인 chronyd 는 명령행 유틸리티 chronyc 에서 모니터링하고 제어할 수 있습니다. 이 유틸리티는 여러 명령을 입력하여 chronyd 의 현재 상태를 쿼리하고 구성을 변경할 수 있는 명령 프롬프트를 제공합니다. 기본적으로 chronyd 는 chronyc 의 로컬 인스턴스의 명령만 허용하지만 원격 호스트에서 모니터링 명령도 허용하도록 구성할 수 있습니다. 원격 액세스는 제한되어야 합니다.

27.2. CHRONYC를 사용하여 CHRONYD 제어

이 섹션에서는 chronyc 명령줄 유틸리티를 사용하여 chronyd 를 제어하는 방법을 설명합니다.

절차

1. *대화형 모드에서 명령행 유틸리티 chronyc 를 사용하여 chronyd 의 로컬 인스턴스를 변경하려면 root 로 다음 명령을 입력합니다.*

```
# chronyc
```

restricted 명령 중 일부를 사용할 경우 chronyc 를 root 로 실행해야 합니다.

chronyc 명령 프롬프트가 다음과 같이 표시됩니다.

```
chronyc>
```

2. *모든 명령을 나열하려면 help 를 입력합니다.*

3. *또는 다음과 같이 명령과 함께 호출되는 경우 비대화형 명령 모드에서 유틸리티를 호출할 수도 있습니다.*

```
chronyc command
```

참고

***chronyc** 를 사용한 변경 사항은 영구적이 아니며 **chronyd** 를 다시 시작한 후 손실됩니다. 영구 변경 사항은 **/etc/chrony.conf** 를 수정합니다.*

28장. CHRONY 사용

다음 섹션에서는 **chronyd** 를 설치, 시작 및 중지하고 **chrony** 가 동기화되었는지 확인하는 방법에 대해 설명합니다. 섹션은 시스템 **Clock**을 수동으로 조정하는 방법도 설명합니다.

28.1. CHRONY 관리

다음 절차에서는 **chronyd** 의 상태를 설치, 시작, 중지 및 확인하는 방법을 설명합니다.

절차

1. **chrony** 제품군은 기본적으로 **Red Hat Enterprise Linux**에 설치됩니다. 이 파일을 확인하려면 **root** 로 다음 명령을 실행하십시오.

```
# dnf install chrony
```

chrony 데몬의 기본 위치는 **/usr/sbin/chronyd** 입니다. 명령행 유틸리티는 **/usr/bin/chronyc** 에 설치됩니다.

2. **chronyd** 의 상태를 확인하려면 다음 명령을 실행합니다.

```
$ systemctl status chronyd
chronyd.service - NTP client/server
  Loaded: loaded (/usr/lib/systemd/system/chronyd.service; enabled)
  Active: active (running) since Wed 2013-06-12 22:23:16 CEST; 11h ago
```

3. **chronyd** 를 시작하려면 **root** 로 다음 명령을 실행합니다.

```
# systemctl start chronyd
```

시스템을 시작할 때 **chronyd** 가 자동으로 시작되도록 하려면 **root** 로 다음 명령을 실행합니다.

```
# systemctl enable chronyd
```

4. **chronyd** 를 중지하려면 **root** 로 다음 명령을 실행합니다.

```
# systemctl stop chronyd
```

시스템 시작 시 **chronyd** 가 자동으로 시작되지 않도록 하려면 **root** 로 다음 명령을 실행합니다.

```
# systemctl disable chronyd
```

28.2. CHRONY가 동기화되었는지 확인

다음 절차에서는 **chrony** 가 **tracking,sources** 및 **sourcestats** 명령을 사용하는 것과 동기화되었는지 확인하는 방법을 설명합니다.

절차

1. **chrony** 추적을 확인하려면 다음 명령을 실행합니다.

```
$ chronyc tracking
Reference ID    : CB00710F (foo.example.net)
Stratum         : 3
Ref time (UTC)  : Fri Jan 27 09:49:17 2017
System time     :  0.000006523 seconds slow of NTP time
Last offset     : -0.000006747 seconds
RMS offset      : 0.000035822 seconds
Frequency       : 3.225 ppm slow
Residual freq   : 0.000 ppm
Skew            : 0.129 ppm
Root delay      : 0.013639022 seconds
Root dispersion : 0.001100737 seconds
Update interval : 64.2 seconds
Leap status     : Normal
```

2. **source** 명령은 **chronyd** 가 액세스 중인 현재 시간 소스에 대한 정보를 표시합니다. **chrony** 소스를 확인하려면 다음 명령을 실행합니다.

```
$ chronyc sources
 210 Number of sources = 3
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
====
#* GPS0                          0   4   377    11   -479ns[ -621ns] /-  134ns
^? a.b.c                        2   6   377    23   -923us[ -924us] +/-   43ms
^ d.e.f                         1   6   377    21  -2629us[-2619us] +/-   86ms
```

선택적 인수 **-v**를 지정할 수 있습니다. 즉 자세한 내용은 다음과 같습니다. 이 경우 추가 주석

줄은 열의 의미를 상기시키는 것으로 표시됩니다.

3. **sourcestats** 명령은 **chronyd** 에서 현재 검사 중인 각 소스에 대한 드리프트 비율 및 오프셋 추정 프로세스에 대한 정보를 표시합니다. **chrony** 소스 통계를 확인하려면 다음 명령을 실행합니다.

```
$ chronyc sourcestats
210 Number of sources = 1
Name/IP Address            NP  NR  Span  Frequency  Freq Skew  Offset  Std Dev
==============================================================================
abc.def.ghi                11   5   46m     -0.001      0.045     1us    25us
```

선택적 인수 **-v** 를 지정할 수 있습니다. 즉 자세한 내용은 다음과 같습니다. 이 경우 추가 주석 줄은 열의 의미를 상기시키는 것으로 표시됩니다.

추가 리소스

- **chronyc(1)** 도움말 페이지

28.3. 시스템 시계 수동 조정

다음 절차에서는 시스템 **Clock**을 수동으로 조정하는 방법을 설명합니다.

절차

1. 시스템 클럭을 즉시 단계적으로 조정하려면 슬래핑하여 진행 중인 조정을 바이패스하고 **root** 로 다음 명령을 실행합니다.

```
# chronyc makestep
```

rtcfile 지시문을 사용하는 경우 실시간 클럭을 수동으로 조정할 수 없습니다. 임의의 조정으로 **chrony**의 실시간 클럭 드리프트가 발생하는 비율을 측정해야 합니다.

28.4. 격리된 네트워크에서 시스템의 CHRONY 설정

인터넷에 연결되지 않은 네트워크의 경우 하나의 컴퓨터가 마스터 **timeserver**로 선택됩니다. 다른 컴퓨터는 마스터 또는 클라이언트의 직접 클라이언트입니다. 마스터에서 드리프트 파일은 시스템 클럭의 평균 드리프트 속도로 수동으로 설정해야 합니다. 마스터가 재부팅되면 주변 시스템에서 시간을 가져오고

*평균을 계산하여 시스템 클럭을 설정합니다. 그런 다음 드리프트 파일에 따라 조정을 다시 시작합니다. **settime** 명령을 사용하면 드리프트 파일이 자동으로 업데이트됩니다.*

*다음 절차에서는 격리된 네트워크에서 시스템에 대한 **chrony** 를 설정하는 방법을 설명합니다.*

절차

1. *루트 로서 실행 중인 텍스트 편집기를 사용하여 마스터로 선택한 시스템에서 다음과 같이 **/etc/chrony.conf** 를 편집합니다.*

```
driftfile /var/lib/chrony/drift
commandkey 1
keyfile /etc/chrony.keys
initstepslew 10 client1 client3 client6
local stratum 8
manual
allow 192.0.2.0
```

*여기서 **192.0.2.0** 은 클라이언트가 연결할 수 있는 네트워크 또는 서브넷 주소입니다.*

2. *마스터의 클라이언트가 **root** 로 실행되는 텍스트 편집기를 사용하여 마스터의 클라이언트를 지시하도록 선택한 시스템에서 다음과 같이 **/etc/chrony.conf** 를 편집합니다.*

```
server master
driftfile /var/lib/chrony/drift
logdir /var/log/chrony
log measurements statistics tracking
keyfile /etc/chrony.keys
commandkey 24
local stratum 10
initstepslew 20 master
allow 192.0.2.123
```

*여기서 **192.0.2.123** 은 마스터의 주소이며 **master** 는 마스터의 호스트 이름입니다. 이 구성을 사용하는 클라이언트는 다시 시작하면 마스터를 재동기화합니다.*

*마스터의 클라이언트가 아닌 클라이언트 시스템에서 **/etc/chrony.conf** 파일은 로컬 및 허용 지시문을 생략해야 한다는 점을 제외하고 동일해야 합니다.*

*격리된 네트워크에서 로컬 참조 모드를 활성화하는 **local** 지시문을 사용하면 로컬 참조 모드를 활성화하여 **NTP** 서버로 작동하는 **chronyd** 가 동기화되지 않았거나 시계의 마지막 업데이트가 오래 전에 발생*

했습니다.

네트워크에 있는 여러 서버가 동일한 로컬 구성을 사용하고 두 서버를 폴링하는 클라이언트의 혼동 없이 서로 동기화되도록 하려면 고립 모드를 활성화하는 로컬 지시문의 분리 옵션을 사용합니다. 다른 모든 서버를 로컬로 폴링하도록 각 서버를 구성해야 합니다. 이렇게 하면 참조 ID가 가장 작은 서버만 로컬 참조가 활성 상태이고 다른 서버가 동기화됩니다. 서버가 실패하면 다른 서버가 대신합니다.

28.5. 원격 모니터링 액세스 구성

Chronyc 는 다음 두 가지 방법으로 chronyd 에 액세스할 수 있습니다.

- *인터넷 프로토콜, IPv4 또는 IPv6.*
- *UNIX 도메인 소켓 - 루트 또는 chrony 사용자가 로컬로 액세스할 수 있습니다.*

기본적으로 chronyc 는 Unix 도메인 소켓에 연결됩니다. 기본 경로는 /var/run/chrony/chronyd.sock 입니다. 이 연결에 실패하면 예를 들어 chronyc 가 루트가 아닌 사용자에서 실행되는 경우 chronyc 는 127.0.0.1에 연결을 시도한 다음 ::1을 시도합니다.

chronyd 의 동작에 영향을 미치지 않는 다음 모니터링 명령만 네트워크에서 허용됩니다.

- *activity*
- *수동 목록*
- *rtcdata*
- *smoothing*
- *소스*

- ***sourcestats***
- ***tracking***
- ***waitsync***

***chronyd** 가 이러한 명령을 허용하는 호스트 세트는 **chronyd** 의 설정 파일에서 **cmdallow** 지시문 또는 **chronyc** 의 **cmdallow** 명령을 사용하여 구성할 수 있습니다. 기본적으로 이 명령은 **localhost(127.0.0.1** 또는 **::1)**에서만 사용할 수 있습니다.*

*다른 모든 명령은 **Unix** 도메인 소켓을 통해서만 허용됩니다. 네트워크를 통해 전송되면 **chronyd** 는 **localhost**에서 가져온 경우에도 **Not authorized error**로 응답합니다.*

*다음 절차에서는 **chronyc** 를 사용하여 **chronyd**에 원격으로 액세스하는 방법을 설명합니다.*

절차

1. */etc/chrony.conf** 파일에 다음을 추가하여 **IPv4** 및 **IPv6** 주소 모두에서 액세스를 허용합니다.*

```
bindcmdaddress 0.0.0.0
```

또는

```
bindcmdaddress ::
```

2. ***cmdallow** 지시문을 사용하여 원격 **IP** 주소, 네트워크 또는 서브넷의 명령을 허용합니다.*

*/etc/chrony.conf** 파일에 다음 내용을 추가합니다.*

```
cmdallow 192.168.1.0/24
```

3. *방화벽에서 포트 **323**을 열어 원격 시스템에서 연결합니다.*

```
#  firewall-cmd --zone=public --add-port=323/udp
```

-

선택적으로 **--permanent** 옵션을 사용하여 포트 **323**을 영구적으로 열 수 있습니다.

```
# firewall-cmd --permanent --zone=public --add-port=323/udp
```

4.

323 포트를 영구적으로 여는 경우 방화벽 구성을 다시 로드합니다.

```
firewall-cmd --reload
```

추가 리소스

- **chrony.conf(5)** 도움말 페이지

28.6. RHEL 시스템 역할을 사용하여 시간 동기화 관리

timesync 역할을 사용하여 여러 대상 시스템에서 시간 동기화를 관리할 수 있습니다. **timesync** 역할은 **PTP** 도메인에서 시스템 클럭을 **NTP** 서버 또는 할머마스터와 동기화하기 위해 **NTP** 또는 **PTP** 슬레이브로 작동하도록 **NTP** 또는 **PTP** 구현을 설치하고 구성합니다.

주의

timesync 역할은 관리 호스트에서 지정된 또는 감지된 공급자 서비스의 구성을 대체합니다. 이전 설정은 역할 변수에 지정되지 않은 경우에도 손실됩니다. **timesync_ntp_provider** 변수가 정의되지 않은 경우 유일한 보존 설정은 공급자를 선택하는 것입니다.

다음 예제에서는 서버 풀 하나만 있는 상황에서 **timesync** 역할을 적용하는 방법을 보여줍니다.

예 28.1. 예제 Playbook은 단일 서버 풀에 timesync 역할을 적용

```
---
- hosts: timesync-test
  vars:
    timesync_ntp_servers:
      - hostname: 2.rhel.pool.ntp.org
        pool: yes
```

```
      iburst: yes
  roles:
    - rhel-system-roles.timesync
```

timesync 역할 변수에 대한 자세한 참조를 보려면 rhel-system-roles 패키지를 설치하고 /usr/share/doc/rhel-system-roles/timesync 디렉터리의 README.md 또는 README.html 파일을 참조하십시오.

추가 리소스

- *RHEL 시스템 역할 시작하기*

28.7. 추가 리소스

- *chronyc(1) 도움말 페이지*
- *chronyd(8) 도움말 페이지*
- *자주하는 질문*

29장. HW 타임스탬프링이 있는 CHRONY

하드웨어 타임스탬프는 일부 NIC(Network Interface Controller)에서 지원되는 기능으로, 수신 및 발신 패킷의 정확한 타임스탬프를 제공합니다. NTP 타임스탬프는 일반적으로 시스템 클럭을 사용하여 커널 및 chronyd 에 의해 생성됩니다. 그러나 HW 타임스탬프ing이 활성화되면 NIC는 자체 클럭을 사용하여 패킷이 링크 계층 또는 물리적 계층을 입력하거나 나가는 경우 타임스탬프를 생성합니다. NTP 와 함께 사용하면 하드웨어 타임스탬프가 동기화의 정확도를 크게 향상시킬 수 있습니다. 최상의 정확도를 위해 NTP 서버와 NTP 클라이언트는 하드웨어 타임스탬프를 사용해야 합니다. 이상적인 조건에서는 마이크로 초의 정확도가 가능할 수 있습니다.

하드웨어 타임스탬프를 사용하는 시간 동기화를 위한 또 다른 프로토콜은 PTP 입니다.

NTP 와 달리PTP 는 네트워크 스위치 및 라우터의 지원에 의존합니다. 동기화의 최상의 정확도에 도달하려면 PTP 지원 기능이 포함된 스위치 및 라우터가 있는 네트워크에서 PTP 를 사용하고 이러한 스위치 및 라우터가 없는 네트워크에서 NTP 를 선호합니다.

다음 섹션에서는 다음 방법을 설명합니다.

- 하드웨어 타임 스탬프에 대한 지원 확인
- 하드웨어 타임스탬프 활성화
- 클라이언트 폴링 간격 구성
- interleaved 모드 활성화
- 다수의 클라이언트에 대해 서버 구성
- 하드웨어 타임스탬프 확인
- PTP-NTP 브리지 구성

29.1. 하드웨어 타임스탬프에 대한 지원 확인

NTP 를 사용한 하드웨어 타임스탬프를 인터페이스에서 확인하려면 ethtool -T 명령을 사용합니다. ethtool 이 SOF_TIMESTAMPING_TX_HARDWARE 및 SOF_TIMESTAMPING_TX_SOFTWARE 기능 및 HWTSTAMP_FILTER_ALL 필터 모드를 나열하는 경우 NTP 를 사용한 하드웨어 타임 스탬프링에 사용할 수 있습니다.

예 29.1. 특정 인터페이스에서 하드웨어 타임스탬프 지원 확인

```
# ethtool -T eth0
```

출력:

```
Timestamping parameters for eth0:
Capabilities:
        hardware-transmit     (SOF_TIMESTAMPING_TX_HARDWARE)
        software-transmit     (SOF_TIMESTAMPING_TX_SOFTWARE)
        hardware-receive      (SOF_TIMESTAMPING_RX_HARDWARE)
        software-receive      (SOF_TIMESTAMPING_RX_SOFTWARE)
        software-system-clock (SOF_TIMESTAMPING_SOFTWARE)
        hardware-raw-clock    (SOF_TIMESTAMPING_RAW_HARDWARE)
PTP Hardware Clock: 0
Hardware Transmit Timestamp Modes:
        off                   (HWTSTAMP_TX_OFF)
        on                    (HWTSTAMP_TX_ON)
Hardware Receive Filter Modes:
        none                  (HWTSTAMP_FILTER_NONE)
        all                   (HWTSTAMP_FILTER_ALL)
        ptpv1-l4-sync         (HWTSTAMP_FILTER_PTP_V1_L4_SYNC)
        ptpv1-l4-delay-req    (HWTSTAMP_FILTER_PTP_V1_L4_DELAY_REQ)
        ptpv2-l4-sync         (HWTSTAMP_FILTER_PTP_V2_L4_SYNC)
        ptpv2-l4-delay-req    (HWTSTAMP_FILTER_PTP_V2_L4_DELAY_REQ)
        ptpv2-l2-sync         (HWTSTAMP_FILTER_PTP_V2_L2_SYNC)
        ptpv2-l2-delay-req    (HWTSTAMP_FILTER_PTP_V2_L2_DELAY_REQ)
        ptpv2-event           (HWTSTAMP_FILTER_PTP_V2_EVENT)
        ptpv2-sync            (HWTSTAMP_FILTER_PTP_V2_SYNC)
        ptpv2-delay-req       (HWTSTAMP_FILTER_PTP_V2_DELAY_REQ)
```

29.2. 하드웨어 타임스탬프 활성화

하드웨어 타임스탬프를 활성화하려면 /etc/chrony.conf 파일에 hwtimestamp 지시문을 사용합니다. 지시문은 단일 인터페이스를 지정하거나 와일드카드 문자를 사용하여 이를 지원하는 모든 인터페이스에서 하드웨어 타임스탬프를 활성화할 수 있습니다. linuxptp 패키지의 ptp4l 과 같은 다른 애플리케이션이 없는 경우 인터페이스에서 하드웨어 타임스탬프를 사용하는 경우 와일드카드 사양을 사용합니다. chrony 구성 파일에서 여러 hwtimestamp 지시문이 허용됩니다.

예 29.2. hwtimestamp 지시문을 사용하여 하드웨어 타임스탬프 활성화

```
hwtimestamp eth0
hwtimestamp eth1
hwtimestamp *
```

29.3. 클라이언트 폴링 간격 구성

폴링 간격(64-1024초)의 기본 범위는 인터넷 서버에 권장됩니다. 로컬 서버 및 하드웨어 타임스탬프의 경우 시스템 클럭의 오프셋을 최소화하기 위해 더 짧은 폴링 간격을 구성해야 합니다.

/etc/chrony.conf 의 다음 지시문은 1초 폴링 간격을 사용하여 로컬 NTP 서버를 지정합니다.

```
server ntp.local minpoll 0 maxpoll 0
```

29.4. INTERLEAVED 모드 활성화

하드웨어 NTP 어플라이언스가 아니라 일반적인 용도의 컴퓨터에서 chrony 와 같은 소프트웨어 NTP 구현을 실행하는 NTP 서버는 패킷을 보낸 후에만 하드웨어 전송 타임스탬프를 가져옵니다. 이 동작은 서버가 해당하는 패킷에 타임스탬프를 저장하지 않도록 합니다. 전송 후 생성된 전송 타임스탬프를 수신하는 NTP 클라이언트가 /etc/chrony.conf 의 server 지시문에 xleave 옵션을 추가하여 NTP 인터리빙 모드를 사용하도록 클라이언트를 구성합니다.

```
server ntp.local minpoll 0 maxpoll 0 xleave
```

29.5. 다수의 클라이언트에 대한 서버 구성

기본 서버 구성을 사용하면 대부분 수천 개의 클라이언트가 동시에 interleaved 모드를 사용할 수 있습니다. 더 많은 수의 클라이언트에 대해 서버를 구성하려면 /etc/chrony.conf 에서 clientloglimit 지시문을 늘립니다. 이 지시문은 서버에 클라이언트 액세스 로깅을 위해 할당된 최대 메모리 크기를 지정합니다.

```
clientloglimit 100000000
```

29.6. 하드웨어 타임스탬프 확인

인터페이스에서 하드웨어 타임스탬프가 성공적으로 활성화되었는지 확인하려면 시스템 로그를 확인합니다. 로그에 성공적으로 활성화된 하드웨어 타임스탬프가 있는 각 인터페이스에 대해 chronyd 의 메시지가 포함되어야 합니다.

예 29.3. 하드웨어 타임스탬프가 활성화된 인터페이스에 대한 로그 메시지

```
chronyd[4081]: Enabled HW timestamping on eth0
chronyd[4081]: Enabled HW timestamping on eth1
```

chronyd 가 **NTP** 클라이언트 또는 피어로 구성된 경우 **chronyc ntpdata** 명령을 통해 각 **NTP** 소스에 대해 보고된 전송 및 타임스탬프 지정 모드와 시간 초과 모드를 사용할 수 있습니다.

예 29.4. 각 **NTP** 소스에 대한 전송, 타임스탬프 지정 및 인터리빙 모드를 보고합니다.

```
# chronyc ntpdata
```

출력:

```
Remote address  : 203.0.113.15 (CB00710F)
Remote port     : 123
Local address   : 203.0.113.74 (CB00714A)
Leap status     : Normal
Version         : 4
Mode            : Server
Stratum         : 1
Poll interval   : 0 (1 seconds)
Precision       : -24 (0.000000060 seconds)
Root delay      : 0.000015 seconds
Root dispersion : 0.000015 seconds
Reference ID    : 47505300 (GPS)
Reference time  : Wed May 03 13:47:45 2017
Offset          : -0.000000134 seconds
Peer delay      : 0.000005396 seconds
Peer dispersion : 0.000002329 seconds
Response time   : 0.000152073 seconds
Jitter asymmetry: +0.00
NTP tests       : 111 111 1111
Interleaved     : Yes
Authenticated   : No
TX timestamping : Hardware
RX timestamping : Hardware
Total TX        : 27
Total RX        : 27
Total valid RX  : 27
```

예 29.5. NTP 측정의 안정성 보고

```
# chronyc sourcestats
```

하드웨어 타임스탬프를 활성화하면 **NTP** 측정의 안정성이 정상적인 부하에 따라 수십 또는 수백 나노초 내에 있어야 합니다. 이 안정성은 **chronyc sourcestats** 명령 출력의 **Std Dev** 열에 보고됩니다.

출력:

```
210 Number of sources = 1
Name/IP Address            NP  NR  Span  Frequency  Freq Skew  Offset  Std Dev
ntp.local                  12   7   11     +0.000      0.019     +0ns    49ns
```

29.7. PTP-NTP 브리지 구성

PTP 지원이 포함된 스위치 또는 라우터가 없는 네트워크에서 매우 정확한PTP(Precision Time Protocol) 하위 마스터를 사용할 수 있는 경우, 컴퓨터는 PTP 슬레이브 및 stratum-1 NTP 서버로 작동하도록 전용될 수 있습니다. 이러한 컴퓨터에는 두 개 이상의 네트워크 인터페이스가 있어야 하며, 할아버지 마스터와 근접하거나 이에 대한 직접 연결이 있어야 합니다. 이렇게 하면 네트워크에서 매우 정확한 동기화가 수행됩니다.

linuxptp 패키지에서 ptp4l 및 phc2sys 프로그램을 구성하여 PTP 를 사용하여 시스템 클럭을 동기화하도록 하나의 인터페이스를 사용합니다.

다른 인터페이스를 사용하여 시스템 시간을 제공하도록 chronyd 를 구성합니다.

예 29.6. 다른 인터페이스를 사용하여 시스템 시간을 제공하도록 chronyd 구성

```
bindaddress 203.0.113.74
hwtimestamp eth1
local stratum 1
```

30장. CHRONY의 NTP(NETWORK TIME SECURITY) 개요

NTP(Network Time Security)는 주요 클라이언트를 확장하도록 설계된 NTP(Network Time Protocol) 인증 메커니즘입니다. 클라이언트 시스템으로 이동하는 동안 서버 시스템에서 수신한 패킷이 변경되지 않은지 확인합니다. NTP(Network Time Security)에는 서버와 클라이언트 간에 사용되는 암호화 키를 자동으로 생성하는 NTPS-KE(Key Establishment) 프로토콜이 포함되어 있습니다.

30.1. 클라이언트 구성 파일에서 NTP(네트워크 시간 보안) 활성화

기본적으로 NTP(Network Time Security)는 활성화되어 있지 않습니다. /etc/chrony.conf 에서 NTS를 활성화할 수 있습니다. 이를 위해 다음 단계를 수행하십시오.

사전 요구 사항

- NTS를 지원하는 서버

절차

클라이언트 구성 파일에서 다음을 수행합니다.

1. 권장되는 iburst 옵션 외에 the nts 옵션을 사용하여 server를 지정합니다.

```
For example:
server time.example.com iburst nts
server nts.netnod.se iburst nts
server ptbtime1.ptb.de iburst nts
```

2. 시스템 부팅 중에 NTPS-KE(Network Time Security-Key Establishment) 세션을 반복하지 않으려면 chrony.conf 에 다음 행을 추가합니다.

```
ntsdumpdir /var/lib/chrony
```

3. DHCP 에서 제공하는 NTP(Network Time Protocol) 서버와의 동기화를 비활성화하려면 chrony.conf 에서 다음 행을 주석 처리하거나 제거합니다(있는 경우).

```
sourcedir /run/chrony-dhcp
```

4. 변경 사항을 저장하십시오.

5. **chronyd** 서비스를 다시 시작하십시오.

```
systemctl restart chronyd
```

검증

- **NTS** 키가 성공적으로 설정되었는지 확인합니다.

```
# chronyc -N authdata

Name/IP address  Mode KeyID Type KLen Last Atmp  NAK Cook CLen
==========================================================
time.example.com  NTS     1   15  256  33m    0    0    8  100
nts.sth1.ntp.se   NTS     1   15  256  33m    0    0    8  100
nts.sth2.ntp.se   NTS     1   15  256  33m    0    0    8  100
```

KeyID,Type 및 **KLen** 의 값은 **0**이 아니어야 합니다. 값이 **0**이면 **chronyd** 에서 시스템 로그에 오류 메시지가 있는지 확인합니다.

- 클라이언트가 **NTP**를 측정하고 있는지 확인합니다.

```
# chronyc -N sources

MS Name/IP address Stratum Poll Reach LastRx Last sample
=======================================================
time.example.com   3       6   377    45   +355us[ +375us] +/-   11ms
nts.sth1.ntp.se    1       6   377    44   +237us[ +237us] +/-   23ms
nts.sth2.ntp.se    1       6   377    44   -170us[ -170us] +/-   22ms
```

Reach 열에는 **0**이 아닌 값이 있어야 합니다. 이상적인 **377**. 값이 **377**를 거의 얻지 못하거나 **377**를 얻지 못하면 네트워크에서 **NTP** 요청 또는 응답이 손실되고 있음을 나타냅니다.

추가 리소스

- **chrony.conf(5)** 도움말 페이지

30.2. 서버에서 NTP(NETWORK TIME SECURITY) 활성화

자체 **NTP(Network Time Protocol)** 서버를 실행하는 경우 서버 **NTP(Network Time Security)** 지원을 활성화하여 클라이언트가 안전하게 동기화하도록 할 수 있습니다.

NTP 서버가 다른 서버, 즉 **Stratum 1** 서버인 경우 동기화에 **NTS** 또는 대칭 키를 사용해야 합니다.

사전 요구 사항

- **PEM** 형식의 서버 개인 키
- 필요한 중간 인증서가 있는 서버 인증서 **PEM** 형식

절차

1. **chrony.conf**에 개인 키와 인증서 파일을 지정합니다.

```
For example:
ntsserverkey /etc/pki/tls/private/foo.example.net.key
ntsservercert /etc/pki/tls/certs/foo.example.net.crt
```

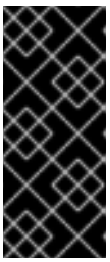
2. 그룹 소유권을 설정하여 **chrony** 시스템 사용자가 키와 인증서 파일을 모두 읽을 수 있는지 확인합니다.

```
For example:
chown :chrony /etc/pki/tls/*/foo.example.net.*
```

3. **the ntsdumpdir /var/lib/chrony** 지시문이 **chrony.conf** 에 있는지 확인합니다.

4. **chronyd** 서비스를 다시 시작하십시오.

```
systemctl restart chronyd
```



중요

서버에 방화벽이 있는 경우 **NTP** 및 **Network Time Security-Key Establishment(NTS-KE)**에 대한 **UDP 123** 및 **TCP 4460** 포트를 모두 허용해야 합니다.

검증

- 다음 명령을 사용하여 클라이언트 시스템에서 빠른 테스트를 수행합니다.

```
$ chronyd -Q -t 3 'server

foo.example.net iburst nts maxsamples 1'
2021-09-15T13:45:26Z chronyd version 4.1 starting (+CMDMON +NTP +REFCLOCK +RTC
+PRIVDROP +SCFILTER +SIGND +ASYNCDNS +NTS +SECHASH +IPV6 +DEBUG)
2021-09-15T13:45:26Z Disabled control of system clock
2021-09-15T13:45:28Z System clock wrong by 0.002205 seconds (ignored)
2021-09-15T13:45:28Z chronyd exiting
```

시스템 클록 잘못된 메시지는 **NTP** 서버가 **NTS-KE** 연결을 수락하고 **NTS** 보호 **NTP** 메시지로 응답하고 있음을 나타냅니다.

- 서버에서 관찰된 **NTS-KE** 연결 및 인증된 **NTP** 패킷을 확인합니다.

```
# chronyc serverstats

NTP packets received       : 7
NTP packets dropped        : 0
Command packets received   : 22
Command packets dropped    : 0
Client log records dropped : 0
NTS-KE connections accepted: 1
NTS-KE connections dropped : 0
Authenticated NTP packets: 7
```

NTS-KE 연결 값이 **0**이 아닌 **NTP** 패킷 필드인 경우 클라이언트가 **NTS-KE** 포트에 연결하고 인증된 **NTP** 요청을 보낼 수 있었습니다.

31장. OPENSSH로 두 시스템 간의 보안 통신 사용

SSH(Secure Shell)는 클라이언트-서버 아키텍처를 사용하여 두 시스템 간에 보안 통신을 제공하고 사용자가 서버 호스트 시스템에 원격으로 로그인할 수 있는 프로토콜입니다. FTP 또는 Telnet과 같은 다른 원격 통신 프로토콜과 달리 SSH는 로그인 세션을 암호화하여 침입자가 연결에서 암호화되지 않은 암호를 수집하지 못하게 합니다.

Red Hat Enterprise Linux에는 일반 openssh 패키지, openssh-server 패키지, openssh-clients 패키지 등 기본 OpenSSH 패키지가 포함되어 있습니다. OpenSSH 패키지에는 OpenSSH가 암호화된 통신을 제공할 수 있는 몇 가지 중요한 암호화 라이브러리를 설치하는 OpenSSL 패키지 openssl-libs가 있어야 합니다.

31.1. SSH 및 OPENSSH

SSH(Secure Shell)는 원격 시스템에 로그인하고 해당 시스템에서 명령을 실행하는 프로그램입니다. SSH 프로토콜은 비보안 네트워크를 통해 신뢰할 수 없는 두 호스트 간에 안전한 암호화된 통신을 제공합니다. 보안 채널을 통해 X11 연결 및 임의의 TCP/IP 포트를 전달할 수도 있습니다.

SSH 프로토콜은 원격 쉘 로그인 또는 파일 복사에 사용할 때 두 시스템 간 통신 가로채기 및 특정 호스트의 가장과 같은 보안 위협을 완화합니다. 이는 SSH 클라이언트와 서버가 디지털 서명을 사용하여 신원을 확인하기 때문입니다. 또한 클라이언트와 서버 시스템 간의 모든 통신이 암호화됩니다.

호스트 키는 SSH 프로토콜에서 호스트를 인증합니다. 호스트 키는 OpenSSH가 처음 설치될 때 또는 호스트가 처음 부팅될 때 자동으로 생성되는 암호화 키입니다.

OpenSSH는 Linux, UNIX 및 유사한 운영 체제에서 지원하는 SSH 프로토콜 구현입니다. OpenSSH 클라이언트와 서버 모두에 필요한 코어 파일을 포함합니다. OpenSSH 제품군은 다음 사용자 공간 툴로 구성됩니다.

- SSH 는 원격 로그인 프로그램(SSH 클라이언트)입니다.
- sshd 는 OpenSSH SSH 데몬입니다.
- SCP 는 안전한 파일 복사 프로그램입니다.

- *SFTP 는 안전한 파일 전송 프로그램입니다.*
- *SSH-agent 는 개인 키를 캐싱하기 위한 인증 에이전트입니다.*
- *ssh-add 는 ssh-agent 에 개인 키 ID를 추가합니다.*
- *SSH-keygen은 ssh 에 대한 인증 키를 생성, 관리 및 변환합니다.*
- *ssh-copy-id 는 원격 SSH 서버의 authorized_keys 파일에 로컬 공개 키를 추가하는 스크립트입니다.*
- *SSH-keyscan 은 SSH 공개 호스트 키를 수집합니다.*

참고

RHEL 9에서는 기본적으로 SFTP(Secure File Transfer Protocol)가 SSH 파일 전송 프로토콜(SCP)으로 교체됩니다. 이는 SCP가 이미 보안 문제를 발생했기 때문입니다(예: CVE-2020-15778).

사용자의 시나리오에서 SFTP를 사용할 수 없거나 호환되지 않는 경우 -O 옵션을 사용하여 원래 SCP/RCP 프로토콜을 강제로 사용할 수 있습니다.

자세한 내용은 Red Hat Enterprise Linux 9에서 OpenSSH SCP 프로토콜 사용 중단 문서를 참조하십시오.

SSH에는 현재 버전 1과 최신 버전 2의 두 가지 버전이 있습니다. RHEL의 OpenSSH 제품군은 SSH 버전 2만 지원합니다. 버전 1에서 알려진 악용에 취약하지 않은 향상된 key-exchange 알고리즘을 가지고 있습니다.

OpenSSH는 RHEL의 핵심 암호화 하위 시스템 중 하나로 시스템 전체 암호화 정책을 사용합니다. 이렇게 하면 기본 구성에서 약한 암호 제품군 및 암호화 알고리즘이 비활성화됩니다. 정책을 수정하려면 관리자는 update-crypto-policies 명령을 사용하여 설정을 조정하거나 시스템 전체 암호화 정책을 수동으로 비활성화해야 합니다.

OpenSSH 제품군은 클라이언트 프로그램(즉, ssh,scp, sftp)과 서버(sshd 데몬)에 대한 두 가지 구성 파일 세트를 사용합니다.

시스템 전체 SSH 구성 정보는 /etc/ssh/ 디렉토리에 저장됩니다. 사용자별 SSH 구성 정보는 사용자 홈 디렉터리의 ~/.ssh/에 저장됩니다. OpenSSH 구성 파일의 자세한 목록은 sshd(8) 도움말 페이지의 FILES 섹션을 참조하십시오.

추가 리소스

- *man -k ssh 명령을 사용하여 나열된 도움말 페이지*
- *시스템 전체 암호화 정책 사용*

31.2. OPENSSH 서버 구성 및 시작

환경 및 OpenSSH 서버를 시작하는 데 필요할 수 있는 기본 구성에는 다음 절차를 사용합니다. 기본 RHEL 설치 후에는 sshd 데몬이 이미 시작되고 서버 호스트 키가 자동으로 생성됩니다.

사전 요구 사항

- *openssh-server 패키지가 설치되어 있어야 합니다.*

절차

1. *현재 세션에서 sshd 데몬을 시작하고 부팅 시 자동으로 시작되도록 설정합니다.*

```
# systemctl start sshd
# systemctl enable sshd
```

2. */etc/ssh/sshd_config 구성 파일의 ListenAddress 지시문의 경우 기본값 0.0.0.0 (IPv4) 또는 :: (IPv6) 이외의 다른 주소를 지정하고 느린 동적 네트워크 구성을 사용하려면 network-online.target 대상 장치에 대한 종속성을 sshd.service 장치 파일에 추가합니다. 이를 위해 다음 내용으로 사용하여 /etc/systemd/system/sshd.service.d/local.conf 파일을 만듭니다.*

```
[Unit]
Wants=network-online.target
After=network-online.target
```

3. */etc/ssh/sshd_config 구성 파일의 OpenSSH 서버 설정이 시나리오 요구 사항을 충족하는지 검토합니다.*

4. *선택적으로 /etc/issue 파일을 편집하여 클라이언트를 인증하기 전에 OpenSSH 서버가 표시하는 시작 메시지를 변경합니다. 예를 들면 다음과 같습니다.*

```
Welcome to ssh-server.example.com
Warning: By accessing this server, you agree to the referenced terms and conditions.
```

Banner 옵션이 /etc/ssh/sshd_config에서 주석 처리되지 않고 해당 값에 /etc/issue가 포함되어 있는지 확인하십시오.

```
# less /etc/ssh/sshd_config | grep Banner
Banner /etc/issue
```

로그인에 성공한 후 표시되는 메시지를 변경하려면 서버에서 /etc/motd 파일을 편집해야 합니다. 자세한 내용은 pam_motd 도움말 페이지를 참조하십시오.

5. *systemd 구성을 다시 로드하고 sshd 를 다시 시작하여 변경 사항을 적용합니다.*

```
# systemctl daemon-reload
# systemctl restart sshd
```

검증

1. *sshd 데몬이 실행 중인지 확인합니다.*

```
# systemctl status sshd
● sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2019-11-18 14:59:58 CET; 6min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
 Main PID: 1149 (sshd)
    Tasks: 1 (limit: 11491)
   Memory: 1.9M
   CGroup: /system.slice/sshd.service
           └─1149 /usr/sbin/sshd -D -oCiphers=aes128-ctr,aes256-ctr,aes128-cbc,aes256-cbc -
oMACs=hmac-sha2-256,>

Nov 18 14:59:58 ssh-server-example.com systemd[1]: Starting OpenSSH server daemon...
```

```
Nov 18 14:59:58 ssh-server-example.com sshd[1149]: Server listening on 0.0.0.0 port 22.
Nov 18 14:59:58 ssh-server-example.com sshd[1149]: Server listening on :: port 22.
Nov 18 14:59:58 ssh-server-example.com systemd[1]: Started OpenSSH server daemon.
```

2. **SSH 클라이언트를 사용하여 SSH 서버에 연결합니다.**

```
# ssh user@ssh-server-example.com
ECDSA key fingerprint is SHA256:dXbaS0RG/UzlTTku8GtXSz0S1++lPegSy31v3L/FAEc.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ssh-server-example.com' (ECDSA) to the list of known hosts.

user@ssh-server-example.com's password:
```

추가 리소스

- **sshd(8) 및 sshd_config(5) 도움말 페이지**

31.3. 키 기반 인증을 위한 OPENSSH 서버 설정

시스템 보안을 강화하려면 **OpenSSH** 서버에서 암호 인증을 비활성화하여 키 기반 인증을 시행합니다.

사전 요구 사항

- **openssh-server** 패키지가 설치되어 있어야 합니다.
- **sshd** 데몬이 서버에서 실행되고 있어야 합니다.

절차

1. 텍스트 편집기에서 **/etc/ssh/sshd_config** 구성을 엽니다. 예를 들면 다음과 같습니다.

```
# vi /etc/ssh/sshd_config
```

2. **PasswordAuthentication** 옵션을 **no**로 변경합니다.

```
PasswordAuthentication no
```

새 기본 설치 이외의 시스템에서 **PubkeyAuthentication no**가 설정되지 않았으며

ChallengeResponseAuthentication 지시문이 **no**로 설정되어 있는지 확인합니다. 콘솔 또는 대역 외 액세스를 사용하지 않고 원격으로 연결하는 경우 암호 인증을 비활성화하기 전에 키 기반 로그인 프로세스를 테스트합니다.

3. **NFS**로 마운트된 홈 디렉토리에서 키 기반 인증을 사용하려면 **use_nfs_home_dirs SELinux** 부울을 활성화합니다.

```
# setsebool -P use_nfs_home_dirs 1
```

4. **sshd** 데몬을 다시 로드하여 변경 사항을 적용합니다.

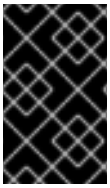
```
# systemctl reload sshd
```

추가 리소스

- **sshd(8), sshd_config(5)** 및 **setsebool(8)** 도움말 페이지

31.4. SSH 키 쌍 생성

이 절차를 사용하여 로컬 시스템에 **SSH** 키 쌍을 생성하고 생성된 공개 키를 **OpenSSH** 서버에 복사합니다. 서버가 적절하게 구성된 경우 암호를 제공하지 않고 **OpenSSH** 서버에 로그인할 수 있습니다.



중요

다음 단계를 **root**로 완료하면 **root**만 키를 사용할 수 있습니다.

절차

1. **SSH** 프로토콜의 버전 **2**에 대한 **ECDSA** 키 쌍을 생성하려면 다음을 수행합니다.

```
$ ssh-keygen -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/home/joesec/.ssh/id_ecdsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/joesec/.ssh/id_ecdsa.
Your public key has been saved in /home/joesec/.ssh/id_ecdsa.pub.
The key fingerprint is:
SHA256:Q/x+qms4j7PCQ0qFd09iZEFHA+SqwBKRNaU72oZfaCI
joesec@localhost.example.com
The key's randomart image is:
```

```
+---[ECDSA 256]---+
|.oo..o=++        |
|.. o .oo .       |
|. .. o. o        |
|....o.+...       |
|o.oo.o +S .      |
|.=.+.   .o       |
|E.*+.  . . .     |
|.=..+ +..  o     |
|  .  oo*+o.      |
+----[SHA256]-----+
```

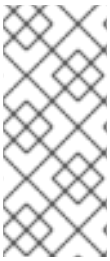
***ssh-keygen -t ed25519** 명령을 입력하여 **ssh-keygen** 명령 또는 **Ed25519** 키 쌍과 함께 **-t rsa** 옵션을 사용하여 **RSA** 키 쌍을 생성할 수도 있습니다.*

2. *공개 키를 원격 머신에 복사하려면 다음을 수행합니다.*

```
$ ssh-copy-id joesec@ssh-server-example.com
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
joesec@ssh-server-example.com's password:
...
Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'joesec@ssh-server-example.com'" and check to make sure that only the key(s) you wanted were added.
```

*세션에서 **ssh-agent** 프로그램을 사용하지 않는 경우 이전 명령은 가장 최근에 수정된 **~/.ssh/id*.pub** 공개 키가 아직 설치되지 않은 경우 공개 키를 복사합니다. 다른 공개 키 파일을 지정하거나 **ssh-agent**로 메모리에 캐시된 키보다 파일의 키 우선 순위를 지정하려면 **ssh-copy-id** 명령을 **-i** 옵션과 함께 사용합니다.*



참고

*시스템을 다시 설치하고 이전에 생성된 키 쌍을 유지하려면 **~/.ssh/** 디렉터리를 백업합니다. 다시 설치한 후 홈 디렉터리로 복사합니다. **root**를 포함하여 시스템의 모든 사용자에 대해 이 작업을 수행할 수 있습니다.*

검증

1. *암호를 제공하지 않고 **OpenSSH** 서버에 로그인합니다.*

```
$ ssh joesec@ssh-server-example.com
Welcome message.
...
```

```
Last login: Mon Nov 18 18:28:42 2019 from ::1
```

추가 리소스

- **ssh-keygen(1)** 및 **ssh-copy-id(1)** 도움말 페이지

31.5. 스마트 카드에 저장된 SSH 키 사용

Red Hat Enterprise Linux를 사용하면 **OpenSSH** 클라이언트의 스마트 카드에 저장된 **RSA** 및 **ECDSA** 키를 사용할 수 있습니다. 다음 절차에 따라 암호 대신 스마트 카드로 인증을 활성화합니다.

사전 요구 사항

- 클라이언트 측에서 **opensc** 패키지가 설치되고 **pcscd** 서비스가 실행 중입니다.

절차

1. **PKCS #11 URI**를 포함하여 **OpenSC PKCS #11** 모듈에서 제공하는 모든 키를 나열하고 출력을 **keys.pub** 파일에 저장합니다.

```
$ ssh-keygen -D pkcs11: > keys.pub
$ ssh-keygen -D pkcs11:
ssh-rsa AAAAB3NzaC1yc2E...KKZMzcQZzx
pkcs11:id=%02;object=SIGN%20pubkey;token=SSH%20key;manufacturer=piv_II?module-
path=/usr/lib64/pkcs11/opensc-pkcs11.so
ecdsa-sha2-nistp256 AAA...J0hkYnnsM=
pkcs11:id=%01;object=PIV%20AUTH%20pubkey;token=SSH%20key;manufacturer=piv_II?
module-path=/usr/lib64/pkcs11/opensc-pkcs11.so
```

2. 원격 서버(**example.com**)에서 스마트 카드를 사용하여 인증을 활성화하려면 공개 키를 원격 서버로 전송합니다. 이전 단계에서 만든 **keys.pub**와 함께 **ssh-copy-id** 명령을 사용하십시오.

```
$ ssh-copy-id -f -i keys.pub username@example.com
```

3. 1단계에서 **ssh-keygen -D** 명령의 출력에서 **ECDSA** 키를 사용하여 **example.com** 에 연결하려면 다음과 같이 키를 고유하게 참조하는 **URI**의 하위 집합만 사용할 수 있습니다.

```
$ ssh -i "pkcs11:id=%01?module-path=/usr/lib64/pkcs11/opensc-pkcs11.so" example.com
Enter PIN for 'SSH key':
[example.com] $
```

4. ~/.ssh/config 파일에서 동일한 URI 문자열을 사용하여 구성을 영구적으로 만들 수 있습니다.

```
$ cat ~/.ssh/config
IdentityFile "pkcs11:id=%01?module-path=/usr/lib64/pkcs11/opensc-pkcs11.so"
$ ssh example.com
Enter PIN for 'SSH key':
[example.com] $
```

OpenSSH는 p11-kit-proxy 래퍼를 사용하고 OpenSC PKCS #11 모듈은 PKCS#11 Kit에 등록되므로 이전 명령을 간소화할 수 있습니다.

```
$ ssh -i "pkcs11:id=%01" example.com
Enter PIN for 'SSH key':
[example.com] $
```

PKCS #11 URI의 id= 부분을 건너뛰면 OpenSSH는 proxy 모듈에서 사용할 수 있는 모든 키를 로드합니다. 이렇게 하면 필요한 입력 횟수가 줄어듭니다.

```
$ ssh -i pkcs11: example.com
Enter PIN for 'SSH key':
[example.com] $
```

추가 리소스

- Fedora 28: OpenSSH에서 스마트 카드 지원 개선
- p11-kit(8), opensc.conf(5), pcscd(8), ssh(1) 및 ssh-keygen(1) 매뉴얼 페이지

31.6. OPENSSH의 보안 강화

다음 팁은 OpenSSH를 사용할 때 보안을 강화하는 데 도움이 됩니다. /etc/ssh/sshd_config OpenSSH 구성 파일의 변경 사항을 적용하려면 sshd 데몬을 다시 로드해야 합니다.

```
# systemctl reload sshd
```

중요

대부분의 보안 강화 구성 변경으로 최신 알고리즘 또는 암호 제품군을 지원하지 않는 클라이언트와의 호환성이 줄어듭니다.

비보안 연결 프로토콜 비활성화

- ***SSH**를 효과적으로 사용하려면 **OpenSSH** 제품군으로 대체되는 안전하지 않은 연결 프로토콜을 사용하지 않도록 합니다. 그렇지 않으면 **Telnet**을 사용하여 로그인할 때 나중에 하나의 세션이 캡처되도록 **SSH**를 사용하여 사용자 암호를 보호할 수 있습니다. 이러한 이유로 **telnet, rsh, rlogin** 및 **ftp**와 같은 비보안 프로토콜을 비활성화하는 것이 좋습니다.*

키 기반 인증 활성화 및 암호 기반 인증 비활성화

- *인증에 대한 암호 비활성화 및 키 쌍만 허용하면 공격 면적이 줄어들고 사용자의 시간도 절약할 수 있습니다. 클라이언트에서 **ssh-keygen** 툴을 사용하여 키 쌍을 생성하고 **ssh-copy-id** 유틸리티를 사용하여 **OpenSSH** 서버의 클라이언트에서 공개 키를 복사합니다. **OpenSSH** 서버에서 암호 기반 인증을 비활성화하려면 **/etc/ssh/sshd_config**를 편집하고 **PasswordAuthentication** 옵션을 **no**로 변경합니다.*

```
PasswordAuthentication no
```

키 유형

- ***ssh-keygen** 명령은 기본적으로 **RSA** 키 쌍을 생성하지만 **-t** 옵션을 사용하여 **ECDSA** 또는 **Ed25519** 키를 생성하도록 지시할 수 있습니다. **ECDSA(Elliptic Curve Digital Signature Algorithm)**는 동등한 대칭 키 힘에서 **RSA**보다 더 나은 성능을 제공합니다. 또한 더 짧은 키를 생성합니다. **Ed25519** 공개 키 알고리즘은 **RSA, DSA** 및 **ECDSA**보다 더 빠르고 안전하며 더 빠릅니다.*

  ***OpenSSH**는 **RSA, ECDSA** 및 **Ed25519** 서버 호스트 키가 누락된 경우 자동으로 생성합니다. **RHEL**에서 호스트 키 생성을 구성하려면 **sshd-keygen@.service** 인스턴스화 서비스를 사용합니다. 예를 들어, **RSA** 키 유형의 자동 생성을 비활성화하려면 다음을 실행합니다.*

```
# systemctl mask sshd-keygen@rsa.service
```

- ***SSH** 연결에 대한 특정 키 유형을 제외하려면 **/etc/ssh/sshd_config**에서 관련 행을 주석 처리하고 **sshd** 서비스를 다시 로드합니다. 예를 들어 **Ed25519** 호스트 키만 허용하려면 다음을 수행합니다.*

```
# HostKey /etc/ssh/ssh_host_rsa_key
# HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
```

기본값 이외의 포트

- *기본적으로 **sshd** 데몬은 **TCP** 포트 **22**에서 수신 대기합니다. 포트를 변경하면 자동화된 네트워크 스캔을 기반으로 하는 공격에 대한 시스템 노출이 줄어들고 따라서 모호성을 통해 보안이 강*

화됩니다. /etc/ssh/sshd_config 구성 파일에서 Port 지시문을 사용하여 포트를 지정할 수 있습니다.

또한 기본이 아닌 포트를 사용할 수 있도록 기본 SELinux 정책을 업데이트해야 합니다. 이렇게 하려면 policycoreutils-python-utils 패키지에서 semanage 툴을 사용합니다.

```
# semanage port -a -t ssh_port_t -p tcp port_number
```

또한 firewalld 구성을 업데이트합니다.

```
# firewall-cmd --add-port port_number/tcp
# firewall-cmd --runtime-to-permanent
```

이전 명령에서 port_number를 Port 지시문을 사용하여 지정된 새 포트 번호로 바꿉니다.

Root 로그인

- PermitRootLogin은 기본적으로 prohibit-password 로 설정됩니다. 이렇게 하면 root로 로그인하는 데 암호를 사용하는 대신 키 기반 인증을 사용하고 무차별 강제 공격을 방지하여 위험을 줄입니다.

경고

관리자가 권한이 있는 명령을 실행하는 사용자를 감사할 수 없기 때문에 root 사용자로 로그인 활성화는 안전한 방법이 아닙니다. 관리 명령을 사용하려면 로그인하고 sudo를 대신 사용합니다.

X 보안 확장 사용

- Red Hat Enterprise Linux 클라이언트의 X 서버는 X 보안 확장을 제공하지 않습니다. 따라서 클라이언트는 X11 전달을 사용하여 신뢰할 수 없는 SSH 서버에 연결할 때 다른 보안 계층을 요청할 수 없습니다. 대부분의 애플리케이션은 이 확장 기능을 사용하여 실행할 수 없습니다.

기본적으로 /etc/ssh/ssh_config.d/05-redhat.conf 파일의 ForwardX11Trusted 옵션은 yes로 설정되어 있으며 ssh -X remote_machine (신뢰할 수 없는 호스트)과 ssh -Y remote_machine (신뢰할 수 있는 호스트) 명령 사이에 차이가 없습니다.

시나리오에 X11 전달 기능이 전혀 필요하지 않은 경우 /etc/ssh/sshd_config 구성 파일의 X11Forwarding 지시문을 no로 설정합니다.

특정 사용자, 그룹 또는 도메인에 대한 액세스 제한

- */etc/ssh/sshd_config 구성 파일 서버의 AllowUsers 및 AllowGroups 지시문을 사용하면 특정 사용자, 도메인 또는 그룹만 OpenSSH 서버에 연결할 수 있습니다. AllowUsers 및 AllowGroups를 결합하여 보다 정확하게 액세스를 제한할 수 있습니다. 예를 들면 다음과 같습니다.*

```
AllowUsers *@192.168.1.*,*@10.0.0.*,!*@192.168.1.2
AllowGroups example-group
```

이전 구성 행은 192.168.1.2 주소가 있는 시스템을 제외하고 192.168.1. 및 10.0.0.* 서브넷의 모든 사용자로부터의 연결을 허용합니다. 모든 사용자는 example-group 그룹에 있어야 합니다. OpenSSH 서버는 다른 모든 연결을 거부합니다.*

허용 목록(허용으로 시작하는 디렉터리)을 사용하는 것은 차단 목록(거부로 시작하는 옵션)을 사용하는 것보다 더 안전합니다. allowlists는 새로운 권한 없는 사용자 또는 그룹도 차단하기 때문입니다.

시스템 전체 암호화 정책 변경

- *OpenSSH는 RHEL 시스템 전체 암호화 정책을 사용하며 기본 시스템 전체 암호화 정책 수준은 현재 위협 모델에 대한 보안 설정을 제공합니다. 암호화 설정을 보다 엄격하게 수행하려면 현재 정책 수준을 변경합니다.*

```
# update-crypto-policies --set FUTURE
Setting system policy to FUTURE
```

- *OpenSSH 서버에 대한 시스템 전체 암호화 정책을 옵트아웃하려면 /etc/sysconfig/sshd 파일에서 CRYPTO_POLICY= 변수로 행의 주석을 해제합니다. 이 변경 후 /etc/ssh/sshd_config 파일의 Ciphers, MACs, KexAlgoritms, GSSAPIKexAlgorithms 섹션에 지정하는 값은 재정의되지 않습니다. 이 작업에는 암호화 옵션 구성에 대한 전문 지식이 필요합니다.*

- *자세한 내용은 보안 강화에서 시스템 전체 암호화 정책 사용을 참조하십시오.*

추가 리소스

- *sshd_config(5), ssh-keygen(1), crypto-policies(7) 및 update-crypto-policies(8) 도움말 페이지*

31.7. SSH 건너뛰기 호스트를 사용하여 원격 서버에 연결

건너뛰기 호스트라고도 하는 중간 서버를 통해 로컬 시스템을 원격 서버에 연결하려면 다음 절차를 사용하십시오.

사전 요구 사항

- *건너뛰기 호스트는 로컬 시스템의 **SSH** 연결을 허용합니다.*
- *원격 서버는 건너뛰기 호스트에서만 **SSH** 연결을 허용합니다.*

절차

1. *로컬 시스템에서 **~/.ssh/config** 파일을 편집하여 건너뛰기를 정의합니다. 예를 들면 다음과 같습니다.*

```
Host jump-server1
 HostName jump1.example.com
```

   - ***Host** 매개 변수는 **ssh** 명령에서 사용할 수 있는 호스트의 이름 또는 별칭을 정의합니다. 이 값은 실제 호스트 이름과 일치할 수 있지만 임의의 문자열일 수도 있습니다.*
   - ***HostName** 매개 변수는 건너뛰기 호스트의 실제 호스트 이름 또는 **IP** 주소를 설정합니다.*

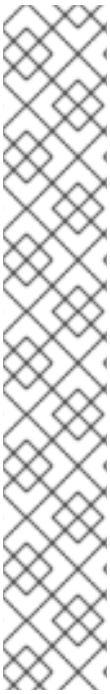
2. ***ProxyJump** 지시문을 사용하여 원격 서버 건너뛰기를 로컬 시스템의 **~/.ssh/config** 파일에 추가합니다. 예를 들면 다음과 같습니다.*

```
Host remote-server
 HostName remote1.example.com
 ProxyJump jump-server1
```

3. *로컬 시스템을 사용하여 이동 서버를 통해 원격 서버에 연결합니다.*

```
$ ssh remote-server
```

*이전 명령은 구성 단계 **1**과 **2**를 생략하면 **ssh -J jump-server1 remote-server** 명령과 동일합니다.*



참고

더 많은 건너뛰기 서버를 지정할 수 있으며 전체 호스트 이름을 제공할 때 구성 파일에 호스트 정의 추가를 건너뛸 수도 있습니다. 예를 들면 다음과 같습니다.

```
$ ssh -J jump1.example.com,jump2.example.com,jump3.example.com
remote1.example.com
```

건너뛰기 서버의 사용자 이름 또는 **SSH** 포트가 원격 서버의 이름과 포트와 다른 경우 이전 명령에서 호스트 이름 전용 표기법을 변경합니다. 예를 들면 다음과 같습니다.

```
$ ssh -J
johndoe@jump1.example.com:75,johndoe@jump2.example.com:75,johndoe@jump3.e
xample.com:75 joesec@remote1.example.com:220
```

추가 리소스

- **ssh_config(5)** 및 **ssh(1)** 도움말 페이지

31.8. SSH-AGENT를 사용하여 SSH 키가 있는 원격 시스템에 연결

SSH 연결을 시작할 때마다 암호를 입력하지 않으려면 **ssh-agent** 유틸리티를 사용하여 개인 **SSH** 키를 캐시할 수 있습니다. 개인 키와 암호는 안전하게 유지됩니다.

사전 요구 사항

- **SSH** 데몬이 실행되고 네트워크를 통해 연결할 수 있는 원격 호스트가 있습니다.
- **IP** 주소 또는 호스트 이름 및 인증 정보를 통해 원격 호스트에 로그인합니다.
- 암호를 사용하여 **SSH** 키 쌍을 생성하고 공개 키를 원격 시스템으로 전송했습니다.

자세한 내용은 **SSH** 키 쌍 생성을 참조하십시오.

절차

1. *선택 사항: 키를 사용하여 원격 호스트에 인증할 수 있는지 확인합니다.*

   a. ***SSH**를 사용하여 원격 호스트에 연결합니다.*

   ```
   $ ssh example.user1@198.51.100.1 hostname
   ```

   b. *개인 키에 대한 액세스 권한을 부여할 키를 만드는 동안 설정한 암호를 입력합니다.*

   ```
   $ ssh example.user1@198.51.100.1 hostname
    host.example.com
   ```

2. ***ssh-agent**를 시작합니다.*

   ```
   $ eval $(ssh-agent)
   Agent pid 20062
   ```

3. ***ssh-agent**에 키를 추가합니다.*

   ```
   $ ssh-add ~/.ssh/id_rsa
   Enter passphrase for ~/.ssh/id_rsa:
   Identity added: ~/.ssh/id_rsa (example.user0@198.51.100.12)
   ```

검증

- *선택 사항: **SSH**를 사용하여 호스트 시스템에 로그인합니다.*

  ```
  $ ssh example.user1@198.51.100.1

  Last login: Mon Sep 14 12:56:37 2020
  ```

  암호를 입력할 필요가 없습니다.

31.9. 추가 리소스

- ***sshd(8), ssh(1), scp(1), sftp(1), ssh-keygen(1), ssh-copy-id(1), ssh_config(5), sshd_config(5), update-crypto-policies(8) 및 crypto-policies(7) 도움말 페이지***

- *OpenSSH Home Page*
- *비표준 구성을 사용하여 애플리케이션 및 서비스에 대한 SELinux 구성*
- *firewalld를 사용하여 네트워크 트래픽 제어*