목차

SUSE Zypper 사용법 3
- ***Zypper*** 3
  - 일반 사용법 3
  - Zypper 명령 3
  - 글로벌 옵션 3
  - 명령 별 옵션 3
  - 인수 4
- ***Zypper 서브커맨드 사용*** 4
- ***Zypper를 통해 소프트웨어 설치 및 제거*** 5
  - 설치 하거나 제거할 패키지 선택 5
  - 패키지 설치 및 제거 결합 7
  - 제거된 패키지의 종속성 정리 7
  - 스크립트에서 자이퍼 사용 8
  - 소스 패키지 설치 또는 다운로드 8
  - 비활성화된 리포지토리에서 패키지 설치 9
  - 유틸리티 9
- ***Zypper로 소프트웨어 업데이트*** 10
  - 필요한 모든 패치 설치 10
  - 목록 패치 11
  - 새 패키지 버전 설치 12
  - 고아 패키지 식별 13
- ***삭제된 파일을 사용하여 프로세스 및 서비스 식별*** 13
- ***Zypper로 리포지토리 관리*** 15
  - 리포지토리 추가 16
  - 리포지토리 갱신 16
  - 리포지토리 제거 17
  - 리포지토리 수정 17
- ***Zypper로 리포지토리 및 패키지 쿼리*** 17
  - 소프트웨어 검색 18
  - 특정 기능 검색 18
  - 패키지 정보 표시 19
- ***Zypper 구성*** 19
- ***문제 해결*** 19
- ***Btrfs 파일 시스템에 지퍼 롤백 기능*** 20

SUSE Zypper 사용법

- https://en.opensuse.org/SDB:Zypper_usage
- https://doc.opensuse.org/documentation/leap/reference/html/book.opensuse.reference/cha-sw-cl.html

Zypper

Zypper는 패키지를 설치, 업데이트 및 제거하기 위한 명령줄 패키지 관리자입니다. 또한 리포지토리를 관리합니다. 원격 소프트웨어 관리 작업을 수행하거나 셸 스크립트에서 소프트웨어를 관리하는 데 특히 유용합니다.

일반 사용법

Zypper의 일반적인 구문은 다음과 입니다.

```
zypper [--global-options] COMMAND  [--command-options] [arguments]
```

Zypper 명령

Zypper를 실행하는 가장 간단한 방법은 그 이름을 입력하고 명령다음에 하는 것입니다. 예를 들어 필요한 모든 패치를 시스템에 적용하려면 다음을 사용하십시오.

```
sudo zypper patch
```

글로벌 옵션

또한 명령 직전에 입력하여 하나 이상의 전역 옵션 중에서 선택할 수 있습니다.

```
sudo zypper --non-interactive patch
```

위의 예에서 `--non-interactive` 이 옵션은 명령이 아무 것도 묻지 않고 실행된다는 것을 의미합니다(기본 답변을 자동으로 적용).

명령 별 옵션

특정 명령에 특정 옵션을 사용하려면 명령 직후에 입력합니다.

```
sudo zypper patch --auto-agree-with-licenses
```

위의 예에서는 `--auto-agree-with-licenses` 라이선스를 확인하라는 요청을 받지 않고 시스템에 필요한 모든 패치를 적용하는 데 사용됩니다. 대신 라이선스가 자동으로 허용됩니다.

인수

일부 명령에는 하나 이상의 인수가 필요합니다. 예를 들어 `install` 명령을 사용할 때는 어떤 패키지 또는 어떤 패키지를 지정해야 합니다.

```
sudo zypper install mplayer
```

일부 옵션에는 단일 인수가 필요합니다. 다음 명령에는 알려진 모든 패턴이 나열됩니다.

```
zypper search -t pattern
```

위의 모든 것을 결합할 수 있습니다. 예를 들어 다음 명령은 `mc` 및 `vim` 패키지를 `factory`레포지토리에서 설치합니다.

```
sudo zypper -v install --from factory mc vim
```

이 `--from` 옵션은 지정된 리포지토리에서 패키지를 요청하는 동안 모든 리포지토리를 사용하도록 설정(종속성을 해결하기 위한)을 유지합니다. `--repo` 옵션은 `--from`의 별칭이며 둘 중 하나를 사용할 수 있습니다.

대부분의 `Zypper` 명령은 지정된 명령을 시뮬레이션하는 `dry-run` 옵션이 있습니다. 테스트 목적으로 사용할 수 있습니다.

```
sudo zypper remove --dry-run MozillaFirefox
```

`Zypper`는 글로벌 옵션 `--userdata STRING`을 지원합니다. 이 옵션으로 문자열을 지정할 수 있으며, 이 옵션은 `Zypper`의 로그 파일 및 플러그인(예: Btrfs 플러그인)에 기록됩니다. 로그 파일에서 트랜잭션을 표시하고 식별하는 데 사용할 수 있습니다.

```
sudo zypper --userdata STRING patch
```

Zypper 서브커맨드 사용

`Zypper` 하위 명령은 `zypper_execdir`에 저장되어있는 `/usr/lib/zypper/commands` 와 같은 실행식입니다. `zypper_execdir` 하위 명령을 찾을 수 없는 경우 `Zypper`는 나머지 `$PATH` 에서 자동으로 검색합니다. 이렇게 하면 고유한 로컬 확장을 작성하고 사용자 공간에 저장할 수 있습니다.

Zypper 셸에서 하위 명령을 실행하는경우 전역 Zypper 옵션을 사용하는 것은 지원되지 않습니다.

사용 가능한 하위 명령을 나열합니다.

```
zypper help subcommand
[...]
Available zypper subcommands in '/usr/lib/zypper/commands'

  appstream-cache
  lifecycle
  migration
  search-packages

Zypper subcommands available from elsewhere on your $PATH

  <none>
```

하위 명령에 대한 도움말 화면을 봅니다.

```
zypper help appstream-cache
```

Zypper를 통해 소프트웨어 설치 및 제거

패키지를 설치하거나 제거하려면 다음 명령을 사용합니다.

```
sudo zypper install PACKAGE_NAME
sudo zypper remove PACKAGE_NAME
```

필수 시스템 패키지를 제거하지 마십시오. Glibc, Zypper, Kernel같은 필수 시스템 패키지를 제거하지 마십시오. 제거되면 시스템이 불안정해지거나 작동을 완전히 중단할 수 있습니다.

설치 하거나 제거할 패키지 선택

명령및 로 패키지를 해결하는 방법에는 여러 가지가 있습니다.

- `zypper install`
- `zypper remove`

정확한 패키지 이름으로

```
sudo zypper install MozillaFirefox
```

정확한 패키지 이름 및 버전 번호로

```
sudo zypper install MozillaFirefox-52.2
```

리포지토리 별칭 및 패키지 이름으로

```
sudo zypper install mozilla:MozillaFirefox
```

와일드 카드를 사용하여 패키지 이름으로

이름이 특정 문자열로 시작하거나 끝나는 모든 패키지를 선택할 수 있습니다. 특히 패키지를 제거할 때 와일드 카드를 주의하여 사용하십시오. 다음 명령은 `Moz`로 시작된 모든 패키지를 설치합니다.

```
sudo zypper install 'Moz*'
```

`-debuginfo`의 모든 패키지 제거 문제를 디버깅할 때 프로세스 실행에 대한 자세한 정보를 제공하는 많은 패키지를 일시적으로 설치해야 하는 경우가 있습니다. 디버깅 세션이 끝나고 환경을 정리해야 하는 후 다음을 실행합니다.

```
sudo zypper remove '*-debuginfo'
```

기능별

예를 들어 이름을 모르게 패키지를 설치하려면 기능이 편리합니다. 다음 명령은 `MozillaFirefox`패키지를 설치합니다.

```
sudo zypper install firefox
```

기능, 하드웨어 아키텍처 또는 버전별

기능과 함께 하드웨어 아키텍처 및 버전을 지정할 수 있습니다.

- 원하는 하드웨어 아키텍처의 이름은 전체 중지 후 기능에 추가됩니다. 예를 들어 `AMD64/Intel64` 아키텍처(Zypper에서 지정한 아키텍처 `x86_64`)를 지정하려면 다음을 사용합니다.

```
sudo zypper install 'firefox.x86_64'
```

- 버전은 문자열의 끝에 추가되어야 하며 연산자가 선행해야 합니다.

```
sudo zypper install 'firefox>=74.2'
```

- 하드웨어 아키텍처 및 버전 요구 사항을 결합할 수도 있습니다.

```
sudo zypper install 'firefox.x86_64>=74.2'
```

RPM 파일로 가는 길별

패키지에 대한 로컬 경로 또는 원격 경로를 지정할 수도 있습니다.

```
sudo zypper install /tmp/install/MozillaFirefox.rpm
sudo zypper install http://download.example.com/MozillaFirefox.rpm
```

패키지 설치 및 제거 결합

패키지를 동시에 설치하고 제거하려면 수정자를 사용합니다. `+/-` 기호를 사용하여 `emacs`를 설치함과 동시에 `vim` 을 제거하려면 아래와 같이 사용한다.

```
sudo zypper install emacs -vim
```

`emacs`를 제거하고 동시에 `vim`을 설치

```
sudo zypper remove emacs +vim
```

패키지 이름이 `-`로 시작하는 경우 `-`기호가 명령 옵션으로 해석되는 것을 방지하려면 `--` 뒤에 인수를 사용하십시요

```
sudo zypper install -emacs +vim       # Wrong
sudo zypper install vim -emacs        # Correct
sudo zypper install -- -emacs +vim    # Correct
sudo zypper remove emacs +vim         # Correct
```

제거된 패키지의 종속성 정리

특정 패키지와 함께 지정된 패키지를 제거한 후 불필요한 패키지를 자동으로 제거하려면 다음 `--clean-deps` 옵션을 사용합니다.

```
sudo zypper rm --clean-deps PACKAGE_NAME
```

스크립트에서 자이퍼 사용

기본적으로 Zypper는 선택한 패키지를 설치하거나 제거하기 전에 또는 문제가 발생할 때 확인을 요청합니다. `--non-interactive`옵션을 사용하여 이 동작을 재정의할 수 있습니다. 이 옵션은 다음에서 볼 수 있듯이 실제 `install` `remove` `patch`명령(및)전에 제공되어야 합니다.

```
sudo zypper --non-interactive install PACKAGE_NAME
```

이 옵션을 사용하면 스크립트 및 크론 작업에 Zypper를 사용할 수 있습니다.

소스 패키지 설치 또는 다운로드

패키지의 해당 소스 패키지를 설치하려면 다음을 사용하십시오.

```
zypper source-install PACKAGE_NAME
```

`root`권한으로 실행될 때 소스 패키지를 설치하는 기본 위치는 `/usr/src/packages/` 이고 일반 사용자의 경우 `~/rpmbuild`로 실행됩니다. 이러한 값은 로컬 `rpm` 설정파일에서 변경할 수 있습니다.

이 명령은 지정된 패키지의 빌드 종속성을 설치합니다. 이 것을 원하지 않는 경우 `-D` 스위치를 추가합니다.

```
sudo zypper source-install -D PACKAGE_NAME
```

빌드 종속성만 설치하려면 `-d`사용합니다.

```
sudo zypper source-install -d PACKAGE_NAME
```

물론 리포지토리 목록에서 소스 패키지가 활성화된 저장소가 있는 경우에만 작동합니다(기본적으로 추가되지만 활성화되지 않음).

리포지토리에서 사용할 수 있는 모든 소스 패키지 목록을 가져올 수 있습니다.

```
zypper search -t srcpackage
```

설치된 모든 패키지에 대한 소스 패키지를 로컬 디렉토리에 다운로드할 수도 있습니다. 소스 패키지를 다운로드하려면 다음을 사용하십시오.

```
zypper source-download
```

기본 다운로드 디렉토리는 `/var/cache/zypper/source-download`. `--directory`옵션을 사용하여 변경할 수 있습니다. 아무것도 다운로드하거나 삭제하지 않고 누락되었거나 불필요한 패키지만 표시하려면 `--status`옵션을 사용합니다. 불필요한 소스 패키지를 삭제하려면 `--delete`옵션을 사용합니다. 삭제를 사용하지 않도록 설정하려면 `--no-delete`옵션을 사용합니다.

비활성화된 리포지토리에서 패키지 설치

일반적으로 활성화된 리포지토리에서만 패키지를 설치하거나 새로 고칠 수 있습니다. 이 `--plus-content TAG` 옵션을 사용하면 리포지토리를 새로 고치고 현재 Zypper 세션 중에 일시적으로 활성화하고 완료한 후 비활성화할 리포지토리를 지정하는 데 도움이 됩니다.

예를 들어 `-debuginfo` 나 `-debugsource` 추가 패키지를 제공할 수 있는 리포지토리를 사용하도록 설정하려면 `--plus-content debug` 를 사용합니다. 이 옵션은 여러 번 지정할 수 있습니다. 이러한 디버깅 리포지토리를 일시적으로 활성화하여 특정 `-debuginfo` 패키지를 설치하려면 다음 옵션을 다음과 같이 사용하십시오.

```
sudo zypper --plus-content debug \
   install "debuginfo(build-id)=eb844a5c20c70a59fc693cd1061f851fb7d046f4"
```

`build-id`문자열은 디버그정보 패키지가 누락된 경우 `gdb`에 의해 보고됩니다.

비활성화된 설치 미디어 openSUSE Leap 설치 미디어는 여전히 구성되지만 성공적인 설치 후에도 비활성화됩니다. 이 `--plus-content`옵션을 사용하여 온라인 리포지토리 대신 설치 미디어에서 패키지를 설치할 수 있습니다. `zypper`를 호출하기 전에 컴퓨터 드라이브에 DVD를 삽입하여 미디어를 사용할 수 있는지 확인합니다.

유틸리티

모든 종속성이 여전히 충족되는지 확인하고 누락된 종속성을 복구하려면 다음을 사용하십시오.

```
zypper verify
```

이행해야 하는 종속성 외에도 일부 패키지 **추천** 다른 패키지. 이러한 권장 패키지는 실제로 사용할 수 있고 설치가능한 경우에만 설치됩니다. 권장 패키지를 설치한 후 권장 패키지를 사용할 수 있는 경우(추가 패키지 또는 하드웨어를 추가하여) 다음 명령을 사용합니다.

```
sudo zypper install-new-recommends
```

이 명령은 웹 캠 또는 Wi-Fi 장치를 연결한 후 매우 유용합니다. 사용 가능한 경우 장치 및 관련 소프트웨어에 대한 드라이버를 설치합니다. 드라이버 및 관련 소프트웨어는 특정 하드웨어 종속성이 충족되는 경우에만 설치할 수 있습니다.

Zypper로 소프트웨어 업데이트

Zypper를 사용하여 소프트웨어를 업데이트하는 방법에는 **패치를 설치**하거나 **새 버전의 패키지를 설치**하거나 **전체 배포판을 업데이트**하여 업데이트하는 세 가지 방법이 있습니다. 전체배포판을 업데이트 하려면 `zypper dist-upgrade`를 이용합니다.

필요한 모든 패치 설치

시스템에 적용되는 공식적으로 릴리스된 모든 패치를 설치하려면 다음을 실행하십시오.

```
sudo zypper patch
```

컴퓨터에서 구성된 리포지토리에서 사용할 수 있는 모든 패치는 설치와 관련성이 있는지 확인합니다. 관련성이 있고(`optional` 또는 `feature`로 분류되지 않은 경우) 즉시 설치됩니다. 설치할 패치에 시스템 재부팅이 필요한 변경 사항이 포함된 경우 이전에 경고를 받게 됩니다.

일반 `zypper patch` 명령은 타사 리포지토리의 패치를 적용하지 않습니다. 타사 리포지토리도 업데이트하려면 다음과 같이 `with-update`명령 옵션을 사용합니다.

```
sudo zypper patch --with-update
```

선택적 패치도 설치하려면 다음을 사용하십시오.

```
sudo zypper patch --with-optional
```

특정 `Bugzilla` 문제와 관련된 모든 패치를 설치하려면 다음을 사용하십시오.

```
sudo zypper patch --bugzilla=NUMBER
```

특정 CVE 데이터베이스 항목과 관련된 모든 패치를 설치하려면 다음을 사용하십시오.

```
sudo zypper patch --cve=NUMBER
```

예를 들어 CVE 번호 `CVE-2010-2713` 가 있는 보안 패치를 설치하려면 다음을 실행합니다.

```
sudo zypper patch --cve=CVE-2010-2713
```

Zypper 및 패키지 관리 자체에 영향을 주는 패치만 설치하려면 다음을 사용하십시오.

```
sudo zypper patch --updatestack-only
```

`updatestack-only` 명령 옵션을 사용하면 다른 리포지토리를 업데이트하는 다른 명령 옵션이 삭제됩니다.

목록 패치

패치를 사용할 수 있는지 여부를 알아내기 위해 Zypper는 다음 정보를 볼 수 있습니다.

필요한 패치 수

필요한 패치(시스템에 적용되지만 아직 설치되지 않은 패치)를 나열하려면 다음을 `patch-check` 사용하십시오.

```
zypper patch-check
Loading repository data...
Reading installed packages...
5 patches needed (1 security patch)
```

이 명령은 Zypper 및 패키지 관리 자체에 영향을 주는 패치만 나열하는 `--updatestack-only`옵션과 결합할 수 있습니다.

필요한 패치 목록

필요한 패치(시스템에 적용되지만 아직 설치되지 않은 패치)를 나열하려면 다음을 `list-patches` 사용하십시오.

```
zypper list-patches
Repository | Name               | Category | Severity | Interactive | Status
| S>
-----------+--------------------+----------+----------+-------------+--------
+-->
Update     | openSUSE-2019-828 | security | moderate | ---         | needed
| S>

Found 1 applicable patch:
1 patch needed (1 security patch)
```

`Since` 컬럼 정보는 `Zypper 1.14.36`이상 버전에서만 표시됩니다.

모든 패치 목록

openSUSE Leap의 사용할 수 있는 모든 패치를 나열하려면 이미 설치되어 있는지 또는 설치에 적용하든 관계없이 `zypper patches` 명령을 사용합니다.

특정 문제와 관련된 패치를 나열하고 설치할 수도 있습니다. 특정 패치를 나열하려면 다음 `zypper list-patches` 옵션을 사용하여 명령을 사용합니다.

버그질라 이슈

Bugzilla 문제와 관련된 필요한 모든 패치를 나열하려면 `--bugzilla` 옵션을 사용합니다.

특정 버그에 대한 패치를 나열하려면 `--bugzilla=NUMBER` 버그 번호를 지정할 수도 있습니다. 여러 Bugzilla 문제와 관련된 패치를 검색하려면 버그 번호 사이에 쉼표를 추가합니다.

```
zypper list-patches --bugzilla=972197,956917
```

CVE 번호로

CVE 데이터베이스(공통 취약점 및 노출)의 항목과 관련된 필요한 모든 패치를 나열하려면 `--cve`옵션을 사용합니다.

특정 CVE 데이터베이스 항목에 대한 패치를 나열하려면 CVE 번호를 지정할 수도 있습니다 `--cve=NUMBER`. 여러 CVE 데이터베이스 항목과 관련된 패치를 검색하려면 다음과 같이 CVE 번호 사이에 쉼표를 추가합니다.

```
zypper list-patches --bugzilla=CVE-2016-2315,CVE-2016-2324
```

필요한지 여부에 관계없이 모든 패치를 나열하려면 `--all`옵션을 추가로 사용합니다. 예를 들어 CVE 번호가 할당된 모든 패치를 나열하려면 다음을 사용하십시오.

```
zypper list-patches --all --cve
Issue | No.            | Patch              | Category    | Severity  | Status
------+----------------+--------------------+-------------+-----------+-------
---
cve   | CVE-2019-0287 | SUSE-SLE-Module.. | recommended | moderate  | needed
cve   | CVE-2019-3566 | SUSE-SLE-SERVER.. | recommended | moderate  | not
needed
[...]
```

새 패키지 버전 설치

리포지토리에 새 패키지만 포함되어 있지만 패치를 제공하지 않는 경우 `zypper patch` 아무효과도 표시되지 않습니다. 설치된 모든 패키지를 최신 버전(시스템 무결성 유지 관리)으로 업데이트하려면 다음

을 사용하십시오.

```
sudo zypper update
```

개별 패키지를 업데이트하려면 업데이트 또는 설치 명령으로 패키지를 지정합니다.

```
sudo zypper update PACKAGE_NAME
sudo zypper install PACKAGE_NAME
```

명령을 통해 모든 새 설치 가능한 패키지 목록을 가져올 수 있습니다.

```
zypper list-updates
```

이 명령은 다음 기준과 일치하는 패키지만 나열합니다.

- 이미 설치된 패키지와 같은 공급업체가 동일합니다.
- 이미 설치된 패키지와 동일한 우선 순위를 가진 리포지토리에서 제공되며,
- 설치가능한 경우(모든 종속성이 충족됩니다).

설치 여부에 관계없이 사용 가능한 새 패키지를 가져올 수 있습니다.

```
sudo zypper list-updates --all
```

새 패키지를 설치할 수 없는 이유를 알아두려면 위에서 설명한 대로 `zypper install` 또는 `zypper update` 명령을 사용합니다.

고아 패키지 식별

Zypper에서 리포지토리를 제거하거나 시스템을 업그레이드할 때 일부 패키지가 고아`(Orphan)`상태 인 경우가 있다. 이러한 고아 패키지는 더 이상 활성 저장소에 속하지 않습니다. 다음 명령은 다음과 같은 목록을 제공합니다.

```
sudo zypper packages --orphaned
```

이 목록을 사용하면 패키지가 여전히 필요한지 또는 안전하게 제거할 수 있는지 여부를 결정할 수 있습니다.

삭제된 파일을 사용하여 프로세스 및 서비스 식별

패키지를 패치, 업데이트 또는 제거할 때 업데이트 또는 제거에 의해 삭제된 파일을 계속 사용하는 프로세스가 시스템에 실행 중인 경우가 있습니다. `zypper ps`는 삭제된 파일을 사용하여 프로세스를 나열하는 데 사용합니다. 프로세스가 알려진 서비스에 속하는 경우 서비스 이름이 나열되어 서비스를 쉽게 다

시 시작할 수 있습니다. 기본적으로 `zypper ps`는 테이블로 표시됩니다.

```
zypper ps
PID   | PPID | UID | User  | Command      | Service      | Files
------+------+-----+-------+--------------+--------------+------------------
-
814   | 1    | 481 | avahi | avahi-daemon | avahi-daemon |
/lib64/ld-2.19.s->
      |      |     |       |              |              |
/lib64/libdl-2.1->
      |      |     |       |              |              |
/lib64/libpthrea->
      |      |     |       |              |              |
/lib64/libc-2.19->
[...]
```

- `PID`: 프로세스의 ID
- `PPID`: 부모 프로세스의 ID
- `UID`: 프로세스를 실행하는 사용자의 ID
- `Login`: 프로세스를 실행하는 사용자의 로그인 이름
- `Command`: 프로세스를 실행하는 데 사용되는 명령
- `Service`: 서비스 이름(명령이 시스템 서비스와 연결된 경우에만)
- `Files`: 삭제된 파일 목록

`zypper ps`의 출력 형식은 다음과 같이 제어할 수 있습니다.

zypper ps -s

삭제된 파일이 표시되지 않는 짧은 테이블을 만듭니다.

```
zypper ps -s
PID   | PPID | UID  | User    | Command      | Service
------+------+------+---------+--------------+--------------
814   | 1    | 481  | avahi   | avahi-daemon | avahi-daemon
817   | 1    | 0    | root    | irqbalance   | irqbalance
1567  | 1    | 0    | root    | sshd         | sshd
1761  | 1    | 0    | root    | master       | postfix
1764  | 1761 | 51   | postfix | pickup       | postfix
1765  | 1761 | 51   | postfix | qmgr         | postfix
2031  | 2027 | 1000 | tux     | bash         |
```

zypper ps -ss

시스템 서비스와 연결된 프로세스만 표시합니다.

```
PID   | PPID | UID  | User    | Command      | Service
------+------+------+---------+--------------+--------------
814   | 1    | 481  | avahi   | avahi-daemon | avahi-daemon
817   | 1    | 0    | root    | irqbalance   | irqbalance
1567  | 1    | 0    | root    | sshd         | sshd
1761  | 1    | 0    | root    | master       | postfix
1764  | 1761 | 51   | postfix | pickup       | postfix
1765  | 1761 | 51   | postfix | qmgr         | postfix
```

zypper ps -sss

삭제된 파일만 사용하여 시스템 서비스를 표시합니다.

```

avahi-daemon
irqbalance
postfix
sshd
```

zypper ps --print "systemctl status %s"

다시 시작해야 할 수 있는 서비스에 대한 상태 정보를 검색하는 명령을 표시합니다.

```

systemctl status avahi-daemon
systemctl status irqbalance
systemctl status postfix
systemctl status sshd
```

서비스 취급에 대한 자세한 내용은 제 10 장, 데몬systemd.

Zypper로 리포지토리 관리

Zypper의 모든 설치 또는 패치 명령은 알려진 리포지토리 목록에 의존합니다. 시스템에 알려진 모든 리포지토리를 나열하려면 명령을 사용합니다.

```

zypper repos
```

결과는 다음 출력과 유사하게 보입니다.

예 2.1: 자이퍼-알려진 저장소 목록

```

zypper repos
#  | Alias                  | Name                | Enabled | GPG Check |
Refresh
---+-----------------------+------------------+----------+-----------+--------
-
 1 | Leap-15.1-Main        | Main (OSS)       | Yes      | (r ) Yes  | Yes
 2 | Leap-15.1-Update      | Update (OSS)     | Yes      | (r ) Yes  | Yes
 3 | Leap-15.1-NOSS        | Main (NON-OSS)   | Yes      | (r ) Yes  | Yes
 4 | Leap-15.1-Update-NOSS | Update (NON-OSS) | Yes      | (r ) Yes  | Yes
[...]
```

다양한 명령에 리포지토리를 지정할 때 `zypper repos` 명령 출력에서 별칭, URI 또는 리포지토리 번호를 사용할 수 있습니다. 리포지토리 별칭은 리포지토리 처리 명령에 사용할 저장소 이름의 짧은 버전입니다. 리포지토리 목록을 수정한 후 리포지토리 번호가 변경될 수 있습니다. 별칭은 그 자체로 변경되지 않습니다.

기본적으로 URI 또는 리포지토리의 우선 순위와 같은 세부 정보가 표시되지 않습니다. 다음 명령을 사용하여 모든 세부 정보를 나열합니다.

```
zypper repos -d
```

리포지토리 추가

리포지토리를 추가하려면

```
sudo zypper addrepo URI ALIAS
```

URI는 인터넷 리포지토리, 네트워크 리소스, 디렉토리 또는 CD 또는 DVD일 수 있습니다. ALIAS는 리포지토리의 약자 및 고유 식별자입니다. 고유해야 하는 유일한 예외를 제외하고 자유롭게 선택할 수 있습니다. 이미 사용 중인 별칭을 지정하면 Zypper가 경고를 발행합니다.

리포지토리 갱신

`zypper`로 구성된 리포지토리에서 패키지의 변경 내용을 가져올 수 있습니다. 변경 내용을 가져오려면 다음을 실행합니다.

```
sudo zypper refresh
```

> `zypper` 기본 동작 기본적으로 일부 명령은 자동으로 `refresh` 수행되므로 명령을 명시적으로 실행할 필요가 없습니다.

이 `refresh` 명령을 사용하면 `--plus-content` 옵션을 사용하여 비활성화된 리포지토리에서도 변경 내용을 볼 수 있습니다.

```
sudo zypper --plus-content refresh
```

이 옵션은 리포지토리의 변경 내용을 가져오지만 비활성화된 리포지토리를 동일한 상태로 유지합니다.

리포지토리 제거

목록에서 리포지토리를 제거하려면 삭제할 저장소별칭 또는 리포지토리 수와 함께 `zypper remove` 명령을 사용합니다. 예를 들어 `Leap-42.3-NOSS` 리포지토리를 제거하려면 다음 명령 중 하나를 사용합니다.

```
sudo zypper removerepo 4
sudo zypper removerepo "Leap-42.3-NOSS"
```

리포지토리 수정

`zypper modifyrepo`로 리포지토리를 사용하거나 비활성화합니다. 이 명령을 사용하면 리포지토리의 속성(예: 새로 고침 동작, 이름 또는 우선 순위)을 변경할 수도 있습니다. 다음 명령을 사용하면 명명된 리포지토리가 활성화되고 자동 새로 고침을 켜고 우선 순위를 20으로 설정합니다.

```
sudo zypper modifyrepo -er -p 20 'updates'
```

리포지토리 수정은 단일 리포지토리에 국한되지 않으며 그룹에서도 작동할 수 있습니다.

- `-a`: 모든 리포지토리
- `-l`: 로컬 리포지토리
- `-t`: 원격 저장소
- `-m TYPE`: 특정 유형의 리포지토리(위치 형식 다음 중 하나가 될 수 있습니다 : `http`, `https`, `ftp`, `cd`, `dvd`, `dir`, `file`, `cifs`, `smb`, `nfs`, `hd`, `iso`)

리포지토리 별칭의 이름을 바꾸려면 `renamerepo`명령을 사용합니다. 다음 예제에서 `Mozilla Firefox` 를 `firefox`로 별칭을 변경합니다.

```
sudo zypper renamerepo 'Mozilla Firefox' firefox
```

Zypper로 리포지토리 및 패키지 쿼리

Zypper는 리포지토리 또는 패키지를 쿼리하는 다양한 방법을 제공합니다. 사용 가능한 모든 제품, 패턴, 패키지 또는 패치 목록을 얻으려면 다음 명령을 사용합니다.

```
zypper products
zypper patterns
zypper packages
zypper patches
```

특정 패키지에 대한 모든 리포지토리를 쿼리하려면 `search`를 사용합니다. 특정 패키지에 대한 정보를 얻으려면 `info`명령을 사용합니다.

소프트웨어 검색

`zypper search`명령은 패키지 이름 또는 패키지 요약 및 설명에서 선택적으로 작동합니다. `/`로 둘러싸인 문자열은 정규표현식으로 해석됩니다. 기본적으로 검색은 대/소문자를 무시합니다.

패키지 이름에 `fire`가 포함된 간단한 검색

```
zypper search "fire"
```

정확한 패키지명 `MozillaFirefox`에 대한 간단한 검색

```
zypper search --match-exact "MozillaFirefox"
```

또한 패키지 설명 및 요약검색

```
zypper search -d fire
```

아직 설치되지 않은 패키지만 출력

```
zypper search -u fire
```

`fir`뒤에 `e`를 포함하지 않는 패키지 검색

```
zypper se "/fir[^e]/"
```

특정 기능 검색

특수 기능을 제공하는 패키지를 검색하려면 `what-provides`명령을 사용합니다. 예를 들어 Perl 모듈의 `SVN::Core` 을 제공하는 패키지를 알고 싶다면 다음 명령을 사용합니다.

```
zypper what-provides 'perl(SVN::Core)'
```

`what-provides PACKAGE_NAME`은 `rpm -q —whatprovides PACKAGE_NAME` 과 유사합니다. 하지만 RPM은 RPM 데이터베이스(설치된 모든 패키지의 데이터베이스)만 쿼리할 수 있습니다. 반면, Zypper

는 설치된 리포지토리뿐만 아니라 모든 리포지토리의 기능 공급자에 대해 알려줍니다.

15 이후 버전부터는 아래와 같이 변경되었습니다.

```
zypper search --provides --match-exact <file>
또는 단순히
zypper search --provides <file>
```

패키지 정보 표시

단일 패키지를 쿼리하려면 정확한 패키지 이름을 인수로 `info` 옵션을 사용합니다. 이 명령은 패키지에 대한 자세한 정보가 표시됩니다. 패키지 이름이 리포지토리의 패키지 이름과 일치하지 않는 경우 명령은 패키지가 아닌 일치항목에 대한 자세한 정보를 출력합니다. 특정 형식(`-t`옵션을 사용하여)을 요청하고 형식이 존재하지 않는 경우 명령은 사용 가능한 다른 일치 항목을 출력하지만 자세한 정보는 없습니다.

소스 패키지를 지정하면 명령에 소스 패키지에서 빌드된 이진 패키지가 표시됩니다. 이진 패키지를 지정하면 명령은 이진 패키지를 빌드하는 데 사용되는 소스 패키지를 출력합니다.

또한 패키지 설치시에 어떤것들이 필요/권장되는 것이 있는지 표시하려면 `--requires` 와 `--recommends` 옵션을 사용합니다.

```
zypper info --requires MozillaFirefox
```

Zypper 구성

이제 Zypper에는 구성 파일이 함께 제공되므로 Zypper의 동작(시스템 전체 또는 사용자 별)을 영구적으로 변경할 수 있습니다. 시스템 전체 설정은 `/etc/zypp/zypper.conf`. 사용자별 설정은 `~/.zypper.conf`. `~/.zypper.conf`파일이 존재하지 않는 경우 `/etc/zypp/zypper.conf`파일을 템플릿으로 사용가능합니다. 해당 파일을 `~/.zypper.conf`로 복사하여 설정하면 됩니다. 사용 가능한 옵션에 대한 도움말을 보려면 파일의 주석을 참조하십시오.

문제 해결

구성된 리포지토리에서 패키지에 액세스하는 데 문제가 있는 경우(예: 리포지토리 중 하나에 있는 것을 알고 있더라도 Zypper는 특정 패키지를 찾을 수 없음) 리포지토리를 새로 고치는 것이 도움이 될 수 있습니다.

```
sudo zypper refresh
```

도움이 되지 않으면

```
sudo zypper refresh -fdb
```

이렇게 하면 원시 메타데이터를 강제로 다운로드하는 것을 포함하여 데이터베이스를 완전히 새로 고치고 다시 빌드합니다.

Btrfs 파일 시스템에 지퍼 롤백 기능

Btrfs 파일 시스템이 루트 파티션에 사용되고 **snapper**가 설치되어있는 경우 Zypper는 파일 시스템을 변경하면 자동으로 **snapper**를 호출하여 적절한 파일 시스템 스냅샷을 만듭니다. 이러한 스냅숏은 Zypper의 변경 내용을 되돌리는 데 사용할 수 있습니다.

From:
https://atl.kr/dokuwiki/ - **AllThatLinux!**

Permanent link:

Last update: **2025/03/07 01:43**