

# podman사용하기

## 1. docker없이 컨테이너 실행하기

1. podman-docker 패키지를 통해 docker가 설치되어 있지 않아도 docker 명령어를 실행할 수 있습니다.
2. 사실 커맨드가 매핑도이 있는것이고 실제로는 podman을 실행하게 됩니다.

```
$> yum install podman-docker -y
```

마지막 메타자료 만료확인 1:40:24 이전인: 2022년 02월 07일 (월) 오후 12시 21분 09초.

종속성이 해결되었습니다.

```
=====
쿠러미   구조 버전                      레포지터리   크기
=====
```

설치 중:

```
podman-docker
noarch 3.3.1-9.0.1.module+el8.5.0+20416+d687fed7 ol8_appstream 56 k
```

연결 요약

설치 1 쿠러미

총계 내려받기 크기: 56 k

설치된 크기 : 230

쿠러미 내려받기중:

```
podman-docker-3.3.1-9.0.1.module+el8.5.0+20416+d 288 kB/s | 56 kB   00:00
```

```
=====
합계                               284 kB/s | 56 kB   00:00
```

연결 확인 실행 중

연결 확인에 성공했습니다.

연결 시험 실행 중

연결 시험에 성공했습니다.

연결 실행 중

```
준비 중      :                               1/1
```

```
설치 중      : podman-docker-3.3.1-9.0.1.module+el8.5.0+20416+d687f 1/1
```

```
스크립트릿 실행 중: podman-docker-3.3.1-9.0.1.module+el8.5.0+20416+d687f 1/1
```

[/usr/lib/tmpfiles.d/pesign.conf:1] Line references path below legacy directory /var/run/, updating /var/run/pesign → /run/pesign; please update the tmpfiles.d/ drop-in file accordingly.

```
확인 중      : podman-docker-3.3.1-9.0.1.module+el8.5.0+20416+d687f 1/1
```

설치되었습니다:

```
podman-docker-3.3.1-9.0.1.module+el8.5.0+20416+d687fed7.noarch
```

완료되었습니다!

```
# yum module install container-tools -y
```

마지막 메타자료 만료확인 0:09:24 이전인: 2022년 02월 07일 (월) 오후 03시 57분 13초.

종속성이 해결되었습니다.

```
=====
쿠러미   구조 버전                      레포지터리   크기
=====
```

그룹/모듈 쿠러미 설치:

```
crun      x86_64 1.0-1.module+el8.5.0+20416+d687fed7 ol8_appstream 193 k
python3-podman noarch 3.2.0-2.module+el8.5.0+20416+d687fed7 ol8_appstream 148 k
udica      noarch 0.2.5-2.module+el8.5.0+20416+d687fed7 ol8_appstream 51 k
```

종속 쿠러미 설치 중:

```
python3-pytml noarch 0.1.14-5.git7dea353.el8 ol8_appstream 25 k
python3-pyxdg noarch 0.25-16.el8 ol8_appstream 94 k
```

모듈 프로파일 설치:

```
container-tools/common
```

연결 요약

## 설치 5 꾸러미

총계 내려받기 크기: 510 k

설치된 크기 : 1.6 M

꾸러미 내려받기중:

```
(1/5): crun-1.0-1.module+el8.5.0+20416+d687fed7. 1.6 MB/s | 193 kB   00:00
(2/5): python3-podman-3.2.0-2.module+el8.5.0+204 65 kB/s | 148 kB   00:02
(3/5): python3-pytmol-0.1.14-5.git7dea353.el8.no 10 kB/s | 25 kB   00:02
(4/5): python3-pyxdg-0.25-16.el8.noarch.rpm 39 kB/s | 94 kB   00:02
(5/5): udica-0.2.5-2.module+el8.5.0+20416+d687fe 33 kB/s | 51 kB   00:01
```

합계 133 kB/s | 510 kB 00:03

연결 확인 실행 중

연결 확인에 성공했습니다.

연결 시험 실행 중

연결 시험에 성공했습니다.

연결 실행 중

```
준비 중 : 1/1
설치 중 : python3-pyxdg-0.25-16.el8.noarch 1/5
설치 중 : python3-pytmol-0.1.14-5.git7dea353.el8.noarch 2/5
설치 중 : python3-podman-3.2.0-2.module+el8.5.0+20416+d687fed7 3/5
설치 중 : udica-0.2.5-2.module+el8.5.0+20416+d687fed7.noarch 4/5
설치 중 : crun-1.0-1.module+el8.5.0+20416+d687fed7.x86_64 5/5
스크립트 실행 중: crun-1.0-1.module+el8.5.0+20416+d687fed7.x86_64 5/5
확인 중 : crun-1.0-1.module+el8.5.0+20416+d687fed7.x86_64 1/5
확인 중 : python3-podman-3.2.0-2.module+el8.5.0+20416+d687fed7 2/5
확인 중 : python3-pytmol-0.1.14-5.git7dea353.el8.noarch 3/5
확인 중 : python3-pyxdg-0.25-16.el8.noarch 4/5
확인 중 : udica-0.2.5-2.module+el8.5.0+20416+d687fed7.noarch 5/5
```

설치되었습니다:

```
crun-1.0-1.module+el8.5.0+20416+d687fed7.x86_64
python3-podman-3.2.0-2.module+el8.5.0+20416+d687fed7.noarch
python3-pytmol-0.1.14-5.git7dea353.el8.noarch
python3-pyxdg-0.25-16.el8.noarch
udica-0.2.5-2.module+el8.5.0+20416+d687fed7.noarch
```

완료되었습니다!

3. docker의 socket api가 지원되기 때문에 podman-docker 패키지를 설치하면 /var/run/docker.sock과 /var/run/podman/podman.sock의 링크를 설정하기 때문에 docker-py, docker-compose를 이용한 docker api도 그대로 사용할 수 있습니다.
4. podman에서 지원하지 않는 docker 옵션은 네트워크, 노드, 플러그인, 이름변경, 시크릿, 서비스, 스택, docker swarm이 대상입니다.

## 2. Rootless설정

1. 호스트 컨테이너 스토리지 경로는 사용자 정보에 따라 달라집니다. (root - /var/lib/containers/storage, non root - \$HOME/.local/share/containers/storage)
2. rootless 컨테이너는 1024미만의 포트에 액세스 할 수 없습니다. (tcp/80을 사용하는 apache의 경우 서비스에서는 tcp/80을 표시하지만 서버 외부에서 실제 접속은 불가능)
3. 1024 이하 포트를 사용하려면 커널값 변경을 통해 임시로 사용할 수는 있으나, 프로덕션 환경에서는 사용하는것은 권장되지 않습니다. (테스트 목적)

```
$> echo 80 > /proc/sys/net/ipv4/ip_unprivileged_port_start
```

4. 8.1부터 rootless Containers기능을 사용하면 일반사용자로 컨테이너 실행할 수 있습니다. (rootless는 기본기능)

```
$> podman pull docker.io/library/httpd
Trying to pull docker.io/library/httpd:latest...
Getting image source signatures
Copying blob 5eb5b503b376 done
Copying blob 10c4d45228bf done
Copying blob a43a76ccc967 done
Copying blob 942bd346e7f7 done
Copying blob cdb155854ae6 done
Copying config a8ea074f45 done
Writing manifest to image destination
```

```
Storing signatures
a8ea074f4566addcd01f9745397f32be471df4a4abf200f0f10c885ed14b1d28
[user@test~]$ docker image ls
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.
REPOSITORY          TAG         IMAGE ID      CREATED       SIZE
docker.io/library/httpd latest      a8ea074f4566  11 days ago  148 MB
```

### 3. 레지스트리 구성

1. registries.conf에 컨테이너를 내려받기 위한 레지스트리 목록이 저장되어 있습니다, 해당 파일에서 검색할 레지스트리를 설정 할 수 있습니다.

```
$> vi /etc/containers/registries.conf
...
unqualified-search-registries = ["container-registry.oracle.com", "docker.io", "container.test.com"]
...
```

2. test.com은 TLS 없이 pull 설정

```
$> vi /etc/containers/registries.conf
...
[[registry]]
location="container.test.com"
insecure = true
...
```

3. 특정 레지스트리 검색 차단

```
$> vi /etc/containers/registries.conf
...
[[registry]]
location="container.test.com"
blocked = true
...
```

### 4. 이미지 검사

1. skopeo 명령어를 통해 이미지 정보를 확인할 수 있습니다.
2. httpd container 정보 확인 방법

```
# skopeo inspect docker://docker.io/library/httpd:latest
{
  "Name": "docker.io/library/httpd",
  "Digest": "sha256:5cc947a200524a822883dc6ce6456d852d7c5629ab177dfbf7e38c1b4a647705",
  "RepoTags": [
    "2",
    "2-alpine",
    "2-alpine3.13",
    "2-alpine3.14",
    "2-alpine3.15",
    "2-bullseye",
    "2-buster",
    "2.2",
    "2.2-alpine",
    "2.2.29",
    "2.2.31",
    "2.2.31-alpine",
    "2.2.32",
    "2.2.32-alpine",
    "2.2.34",
    "2.2.34-alpine",
    "2.4",
    "2.4-alpine",
    "2.4-alpine3.13",
    "2.4-alpine3.14",
    "2.4-alpine3.15",
    "2.4-bullseye",
    "2.4-buster",
    "2.4.10",
    "2.4.12",
    "2.4.16",
```

```
"2.4.17",
"2.4.18",
"2.4.20",
"2.4.23",
"2.4.23-alpine",
"2.4.25",
"2.4.25-alpine",
"2.4.27",
"2.4.27-alpine",
"2.4.28",
"2.4.28-alpine",
"2.4.29",
"2.4.29-alpine",
"2.4.32",
"2.4.32-alpine",
"2.4.33",
"2.4.33-alpine",
"2.4.34",
"2.4.34-alpine",
"2.4.35",
"2.4.35-alpine",
"2.4.37",
"2.4.37-alpine",
"2.4.38",
"2.4.38-alpine",
"2.4.39",
"2.4.39-alpine",
"2.4.41",
"2.4.41-alpine",
"2.4.43",
"2.4.43-alpine",
"2.4.46",
"2.4.46-alpine",
"2.4.47",
"2.4.47-alpine",
"2.4.48",
"2.4.48-alpine",
"2.4.48-alpine3.13",
"2.4.48-alpine3.14",
"2.4.48-buster",
"2.4.49",
"2.4.49-alpine",
"2.4.49-alpine3.14",
"2.4.49-buster",
"2.4.50",
"2.4.50-alpine",
"2.4.50-alpine3.14",
"2.4.50-buster",
"2.4.51",
"2.4.51-alpine",
"2.4.51-alpine3.14",
"2.4.51-alpine3.15",
"2.4.51-bullseye",
"2.4.51-buster",
"2.4.52",
"2.4.52-alpine",
"2.4.52-alpine3.15",
"2.4.52-bullseye",
"alpine",
"alpine3.13",
"alpine3.14",
"alpine3.15",
"bullseye",
"buster",
"latest"
],
"Created": "2022-01-26T08:38:51.175633696Z",
"DockerVersion": "20.10.7",
"Labels": null,
```

```
"Architecture": "amd64",
"Os": "linux",
"Layers": [
  "sha256:5eb5b503b37671af16371272f9c5313a3e82f1d0756e14506704489ad9900803",
  "sha256:a43a76ccc96739928c7e884f2210dde01ae5b1fd8822f8cdd56e6ba64ec3125a",
  "sha256:942bd346e7f719d26e022dfc42eea7e1fa5cf9ad60ec80ed0ef79ded05288be6",
  "sha256:cdb155854ae6a9e25834459a7c9dfc7be157a2ebfca5adbdb036aeea43ce3128",
  "sha256:10c4d45228bf56285a1e2c828d60e34d8413ee80e1abd738ef190be843d9dc1e"
],
"Env": [
  "PATH=/usr/local/apache2/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
  "HTTPD_PREFIX=/usr/local/apache2",
  "HTTPD_VERSION=2.4.52",
  "HTTPD_SHA256=0127f7dc497e9983e9c51474bed75e45607f2f870a7675a86dc90af6d572f5c9",
  "HTTPD_PATCHES="
]
```

---

🕒Revision #1

★Created 8 June 2022 03:19:55 by artop0420

✎Updated 24 December 2023 02:58:52 by artop0420