도커 체크포인트 사용하기 (우분투 20.04 기준)

참고: https://criu.org/Docker, https://criu.org/Installation

1.도커에서 실험적기능 활성화

Docker Experimental [edit]

Naturally, Docker wants to manage the full lifecycle of processes running inside its containers, so CRIU should be run by Docker (rather than separately). This feature is available in the *experimental* mode for Docker (since Docker 1.13, so every later version, like Docker 17.03, should work).

To enable experimental features (incl. CRIU), you need to do something like this:

```
echo "{₩"experimental₩": true}" >> /etc/docker/daemon.json
systemctl restart docker
```

2.CRIU 설치를 위한 추가 패키지 설치

Compiler and C Library [edit]

CRIU is mostly written in C and the build system is based on Makefiles. Thus just install standard `gcc` and `make` packages (on Debian use `build-essential`).

For building with 32bit tasks C/R support you will need `libc6-dev-i386`, `gcc-multilib` instead of `gcc`.

Cross-compilation for ARM is also possible.

Protocol Buffers [edit]

CRIU uses the Google Protocol Buffers to read and write images. The `protoc` tool is used at build time and CRIU is linked with the `libprotobuf-c.so`. Also CRIT uses python bindings and the `descriptor.proto` file which typically provided by a distribution's protobuf development package.

RPM packages

`protobuf protobuf-c protobuf-c-devel protobuf-compiler protobuf-devel protobuf-python`

Deb packages

`libprotobuf-dev libprotobuf-c-dev protobuf-c-compiler protobuf-compiler python-protobuf`

Optionally, you may build protobuf from sources.

Other stuff [edit]

- `pkg-config` to check on build library dependencies.
- `python-ipaddress` is used by CRIT to pretty-print IP addresses and is also required by zdtm.py
- `libbsd-devel` (RPM) / `libbsd-dev` (DEB) If available, CRIU will be compiled with `setproctitle()` support and set verbose process titles on service workers.
- `iproute2` version 3.5.0 or higher is needed for dumping network namespaces. The latest one can be cloned from iproute2. It should be compiled and a path to ip set as the `CR_IP_TOOL` variable
- `nftables` (RPM) / `libnftables-dev` (DEB) If available, CRIU will be compiled with nftables C/R support
- `libcap-devel` (RPM) / `libcap-dev` (DEB)
- `libnet-devel libnl3-devel` (RPM) / `libnet1-dev` (DEB) / `libnl-3-dev libnet-dev` (Ubuntu)
- `libaio-devel` (RPM) / `libaio-dev` (DEB) is needed to run tests
- `python2-future` or `python3-future` is now needed for zdtm.py tests launcher

For APT use the `--no-install-recommends` parameter is to avoid asciidoc pulling in a lot of dependencies. Also read about ZDTM test suite if you will run CRIU tests, those sources need other deps.

3.CRIU 소스코드 다운로드

/home/daniel/바탕화면/ 아래에 폴더를 만들고 (한글이 경로명에 있어도 괜찮더라...) git clone https://github.com/checkpoint-restore/criu.git

다운받은 디렉토리로 이동하여서 ./make 실행 (sudo 없어도 괜찮더라)

/bin 디렉토리로 이동해서 심볼릭 링크 만들기: sudo ln -s 원본경로 ./criu

4.CRIU 설치 여부 확인: sudo criu check

```
daniel@dan-vb-base: /bin
daniel@dan-vb-base:/bin$ sudo criu check
Looks good.
daniel@dan-vb-base:/bin$
```

5.도커에서 체크포인트 생성하기

1) 도커 (재)실행

```
daniel@dan-vb-base:~$ docker restart naughty_tereshkova
naughty_tereshkova
daniel@dan-vb-base:~$ docker attach naughty_tereshkova
root@6e9916a34010:/#
```

2) 체크포인트 생성

- 생성하면 도커는 자동으로 실행이 중단됨

-"--leave-running" 옵션을 주면 실행이 중단되지 않음

```
daniel@dan-vb-base:~$ docker checkpoint create naughty_tereshkova check2
check2
daniel@dan-vb-base:~$
```

3) 생성된 체크포인트 확인

daniel@dan-vb-base: ~

```
daniel@dan-vb-base:~$ docker checkpoint ls naughty_tereshkova
CHECKPOINT NAME
check1
check2
daniel@dan-vb-base:~$
```

** 현재 디렉토리에 체크포인트 저장하기 + 도커 정지하지 않기 (체크 포인트를 만들면 root 권한 폴더로 생성되네. 참고: 체크 포인트는 폴더로 생성됨)

$docker checkpoint create naughty_tereshkova check4 --checkpoint-dir=/home/daniel/바탕화면/docker_migration/git-workspace/DockerCheckpoint/ --leave-running

옵션들... (https://github.com/docker/cli/blob/master/experimental/checkpoint-restore.md)

The options for checkpoint create:

```
Usage:  docker checkpoint create [OPTIONS] CONTAINER CHECKPOINT

Create a checkpoint from a running container

  --leave-running=false    Leave the container running after checkpoint
  --checkpoint-dir         Use a custom checkpoint storage directory
```

And to restore a container:

```
Usage:  docker start --checkpoint CHECKPOINT_ID [OTHER OPTIONS] CONTAINER
```