··cgroup·····

·cgroup v2···memory.max····memory.high·····memory.current·····OOM···

····· cgroup v2 ····· cgroup···
····· OOM ·····

····· cgroup ·····

···· PID····· /proc/PID/cgroup · /proc/PID/mountinfo··· unified cgroup v2 ····· cgroup na
mespace ····· namespace ····· /sys/fs/cgroup ·····
·····

·· | ·· | ····

memory.max | cgroup··· | max·····

memory.high | ····· | ····OOM···

memory.current | ·cgroup····· | ·····

memory.events | ···OOM····· | ·····

·····

·····ID···PID·····
·····

····· namespace ····· CGROUP_DIR ·····
····· memory.max····· high·current · events···· current ·····
··

```
from pathlib import Path
import json, sys

mount = Path(sys.argv[1]).resolve()
group = Path(sys.argv[2]).resolve()
if not group.is_relative_to(mount):
    raise SystemExit('··cgroup·····')
if not (group / 'memory.current').is_file():
    raise SystemExit('···memory.current···cgroup v2·memory···')

limits = []
current = group
while current == mount or mount in current.parents:
    limit = current / 'memory.max'
    if limit.is_file():
        value = limit.read_text().strip()
        limits.append({'path':str(current), 'memory.max':value})
    if current == mount:
        break
    current = current.parent

result = {'group':str(group), 'memory.current':(group/'memory.current').read_text().strip(),
          'memory.high':(group/'memory.high').read_text().strip()
          if (group/'memory.high').is_file() else 'unavailable',
          'limits_target_and_ancestors':limits}
for filename in ('memory.events','memory.events.local'):
    path = group / filename
    if path.is_file():
        result[filename] = dict(line.split() for line in path.read_text().splitlines())
print(json.dumps(result, indent=2))
```

·····memory.events ····· oom·oom_kill ·····
···memory.high ·····memory.max ·····
·····

·cgroup·····

·· memory.events ····· cgroup ·····
····· job ID ·····

····· PID····· RSS··· MemAvailable ···/·····RSS ·····memory
.current ·····Docker · OOMKilled ····· cgroup ·····
·····

····· memory.max · memory.high····· memory.high
····· oom_kill·····
····· cgroup ··· namespace·····

····

Linux····cgroup v2·····memory···· https://docs.kernel.org/admin-guide/cgroup-v2.html

Linux····cgroup namespace·/proc···· https://docs.kernel.org/admin-guide/cgroup-v2.html#namespace

Docker·····OOM···· https://docs.docker.com/engine/containers/resource_constraints/