C3, 데이터 처리에서 서빙까지 가능한 하둡 클러스터

DEVIEW 2018

남경완

System&Solution

NAVER

CONTENTS

DEVIEW 2018

1. C3 Hadoop Cluster
2. How to
3. Considerations & Issues
4. Future work

1.
C3 Hadoop Cluster

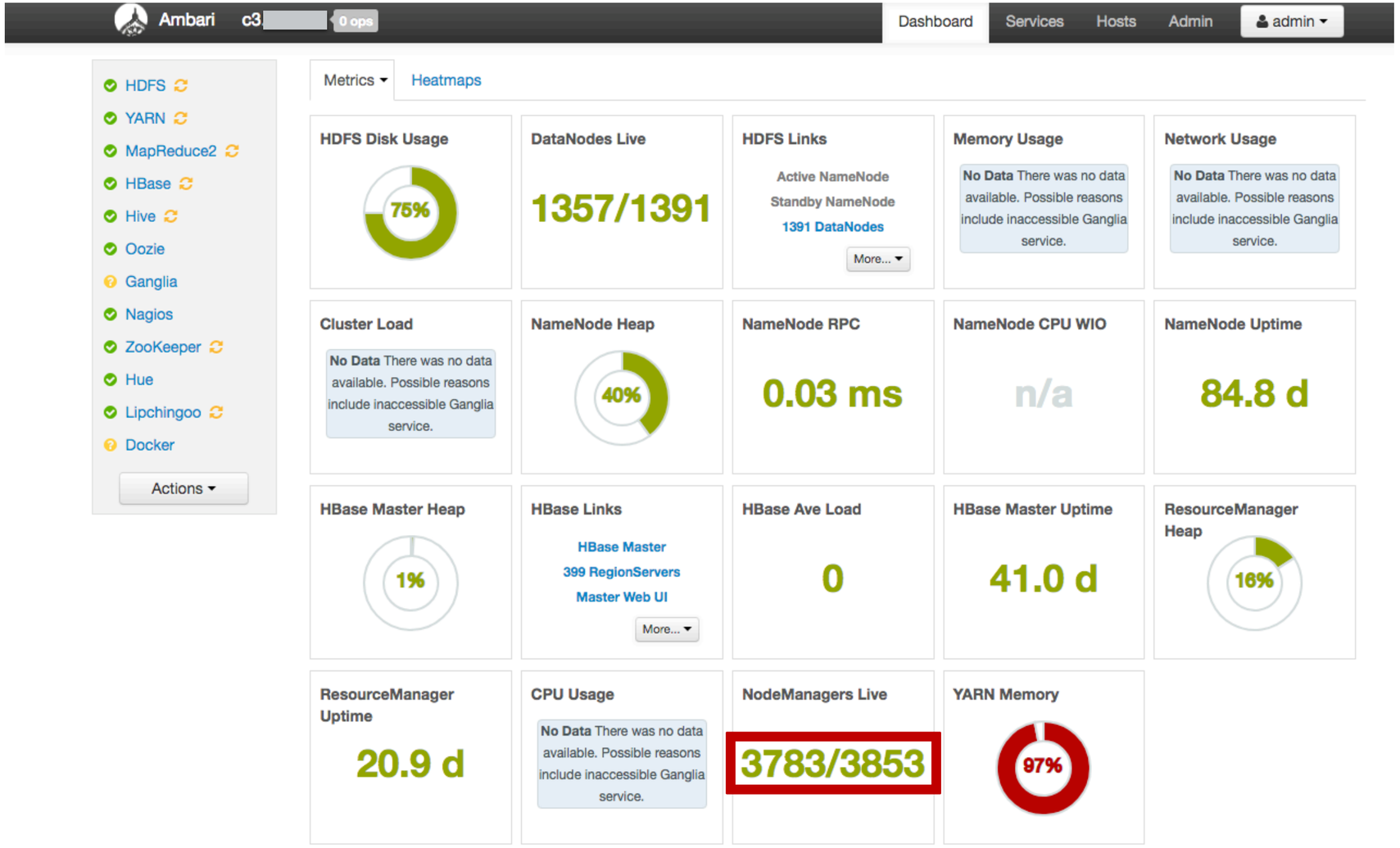
DEVIEW
2018

1.1 C3?

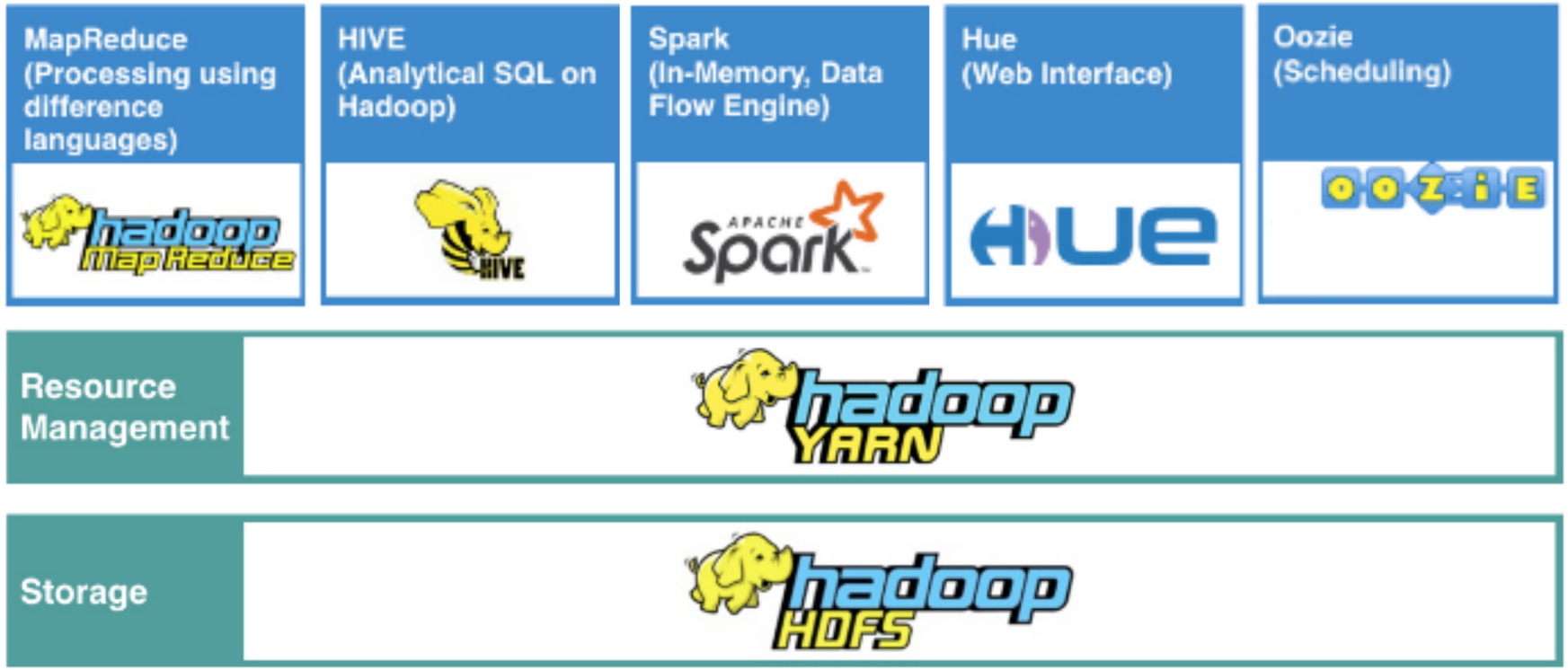
3500+ nodes

500+ users

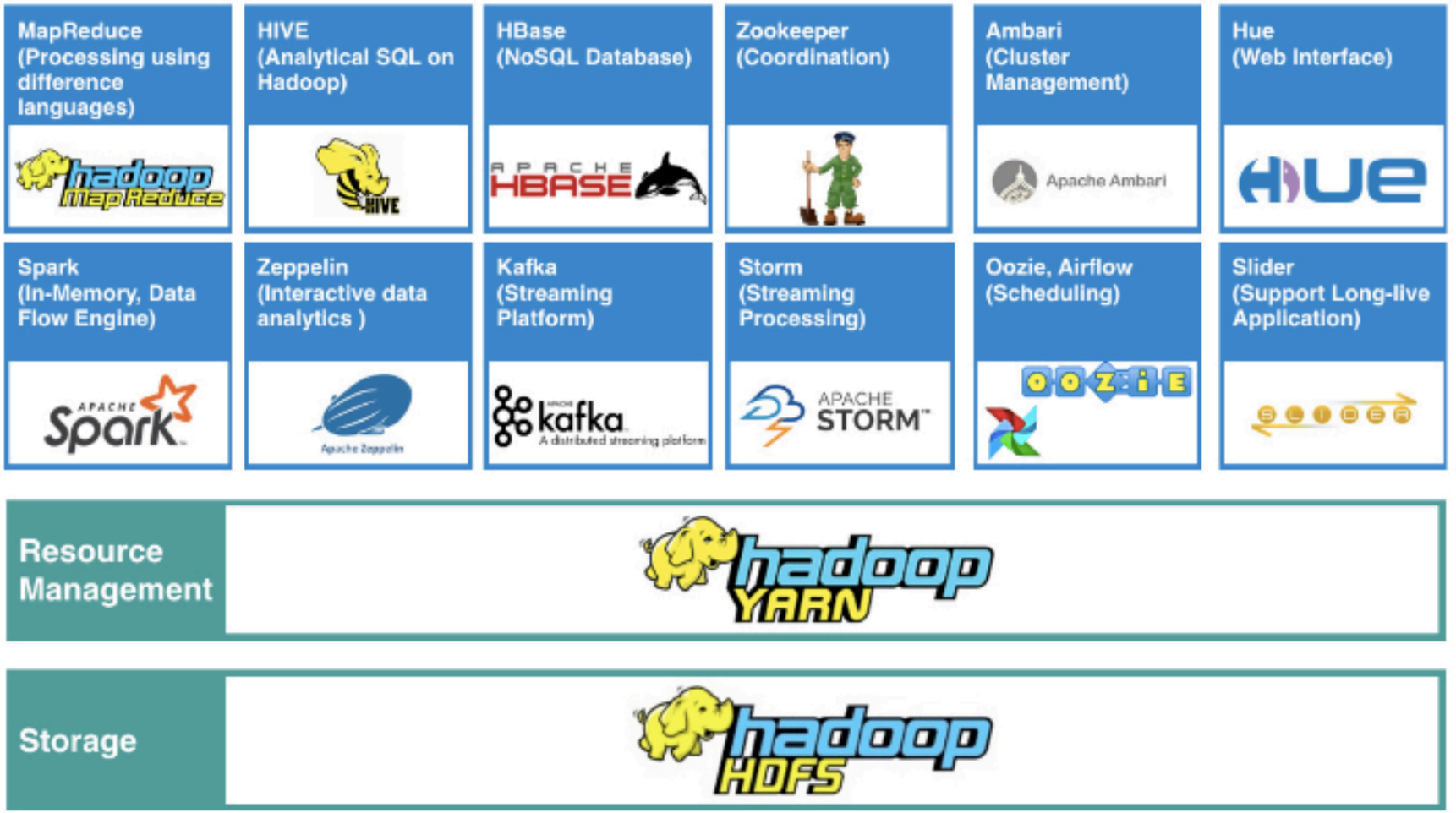
Hadoop 2.7.1 + patch



1.2 C3 v1


MapReduce (Processing using difference languages)
HIVE (Analytical SQL on Hadoop)
Spark (In-Memory, Data Flow Engine)
Hue (Web Interface)
Oozie (Scheduling)
Resource Management
Storage

1.3 C3 v2


MapReduce (Processing using difference languages)
HIVE (Analytical SQL on Hadoop)
HBase (NoSQL Database)
Zookeeper (Coordination)
Ambari (Cluster Management)
Hue (Web Interface)
Spark (In-Memory, Data Flow Engine)
Zeppelin (Interactive data analytics)
Kafka (Streaming Platform)
Storm (Streaming Processing)
Oozie, Airflow (Scheduling)
Slider (Support Long-live Application)
Resource Management
Storage

1.4 C3 현재

DEVIEW 2018

1.5 Why?

DEVIEW 2018

Resource Utilization

- 다양한 작업들의 리소스 공유

Scalability & Flexibility

- 쉽게 확장 가능

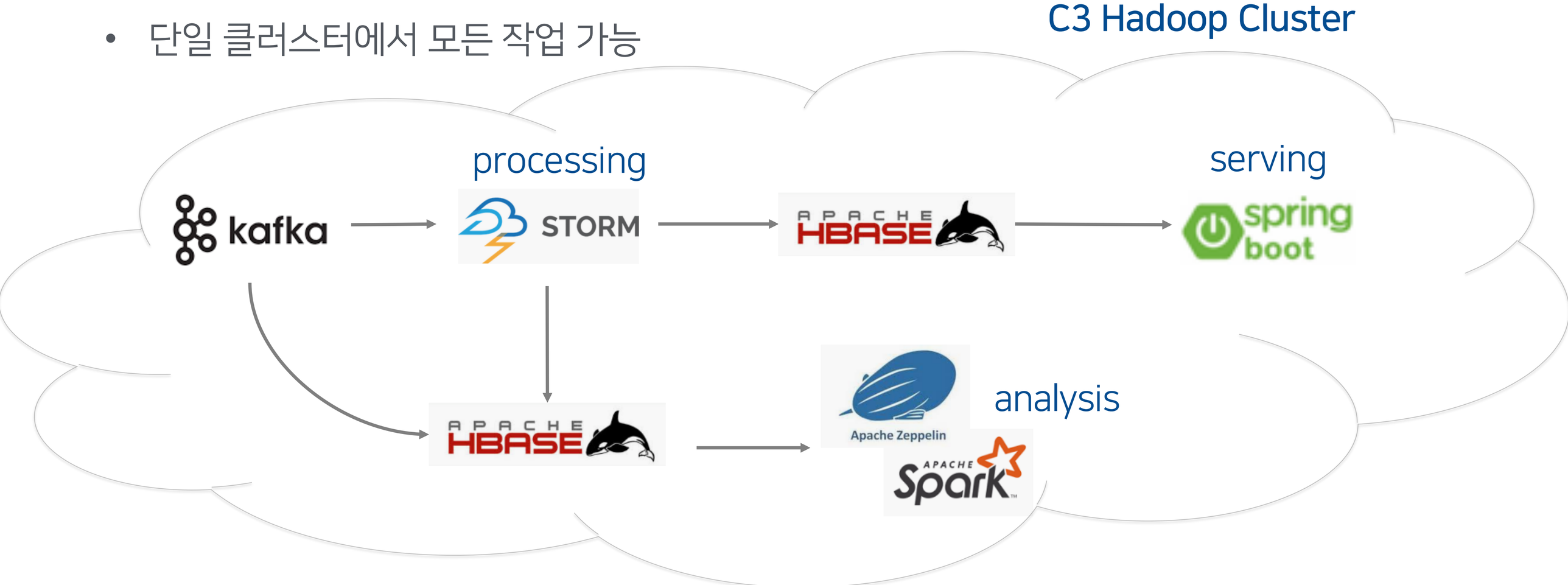
Recovery

- 장애 대응

1.5 Why?

Agility

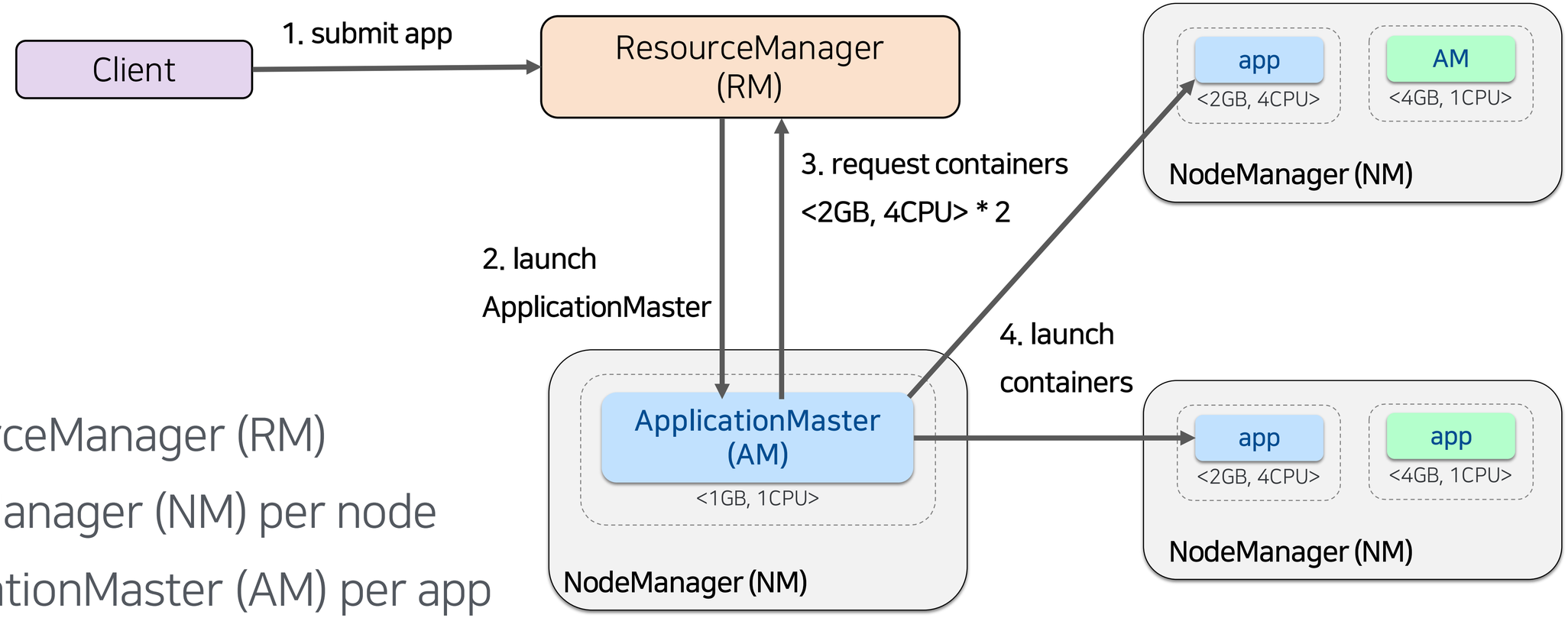
- 필요한 앱을 쉽게 실행하여 비즈니스 로직에 집중
- 단일 클러스터에서 모든 작업 가능



2.
HOW TO

DEVIEW
2018

2.1 Hadoop YARN



ResourceManager (RM)

NodeManager (NM) per node

ApplicationMaster (AM) per app

원하는 앱을 실행하도록
YARN Application 을 구현?

2.2 Apache Slider

DEVIEW 2018

YARN 에서 원하는 앱을 쉽게 실행시킬 수 있는 프레임워크

YARN 컨테이너 환경의 이점 활용

- Availability - 장애시 새로운 컨테이너를 할당받아 자동 복구
- Flexibility – 동적으로 컨테이너를 추가하여 쉽게 확장

Apache Slider 를 이용한 멀티테넌트 하둡 클러스터

- http://deview.kr/2016/schedule#session/168

2.2 Apache Slider

DEVIEW 2018

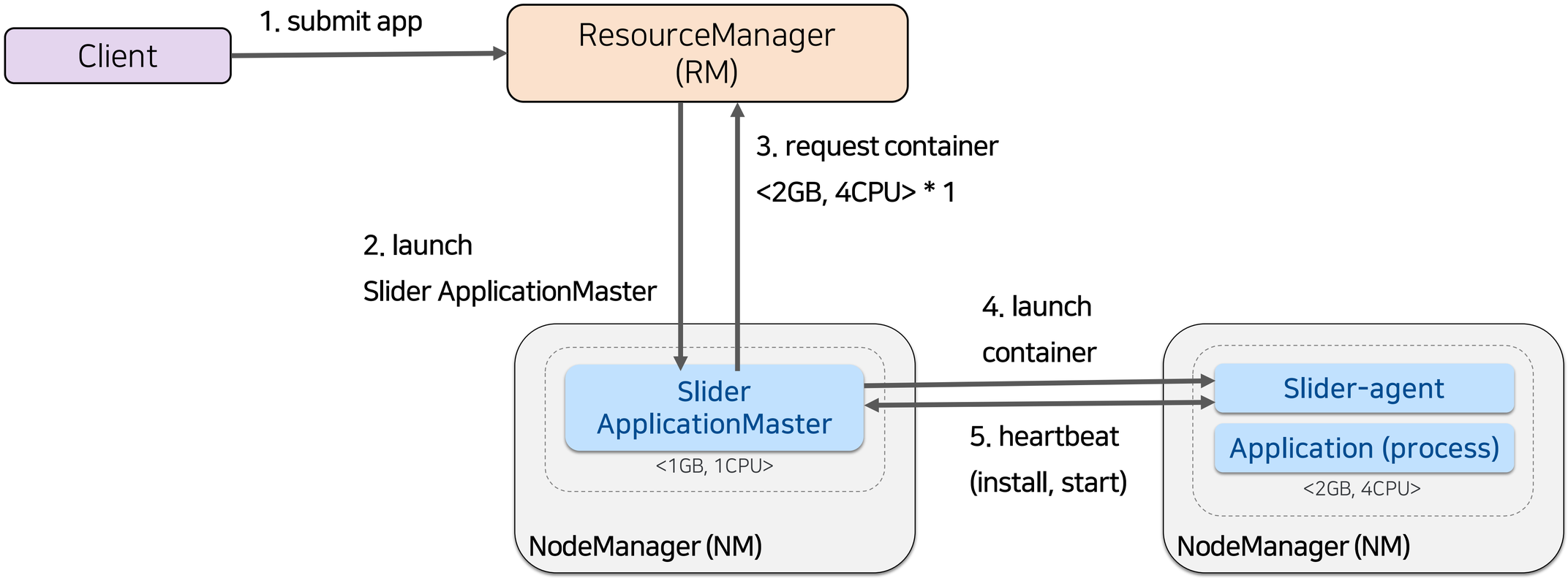
Sliderize

- 앱에 대한 정의 및 설정 파일
- 앱의 패키지 파일
- 앱의 lifecycle 에 대한 python 코드 작성

```
resources-default.json
README.md
appConfig-default.json
metainfo.xml
configuration
package/scripts/hbase_master.py
package/files/hbase-1.2.1.tar.gz
package/templates
```

```python
class HbaseMaster(Script):
  def install(self, env):
    self.install_packages(env)

  def start(self, env):
    import params
    env.set_params(params)
    self.configure(env) # for security

    hbase_service('master', action = 'start')

  def stop(self, env):
    import params
    env.set_params(params)

    hbase_service('master', action = 'stop')

  def status(self, env):
    import status_params
    env.set_params(status_params)
    pid_file = format("{pid_dir}/hbase-{hbase_user}-master.pid")
    check_process_status(pid_file)
```

2.3 Slider on YARN



사용자가 원하는 모든 앱을

Sliderize?

2.4 Docker

DEVIEW 2018

Build, Ship, Run

- 한번 만들면 쉽게 배포하고 실행
- os, library 등으로 인한 의존성 문제 해결
- 이미 누군가 잘 만든 이미지 활용 가능

대부분의 개발자는 Sliderize 보다 Dockerize 익숙

2.5 Support docker container

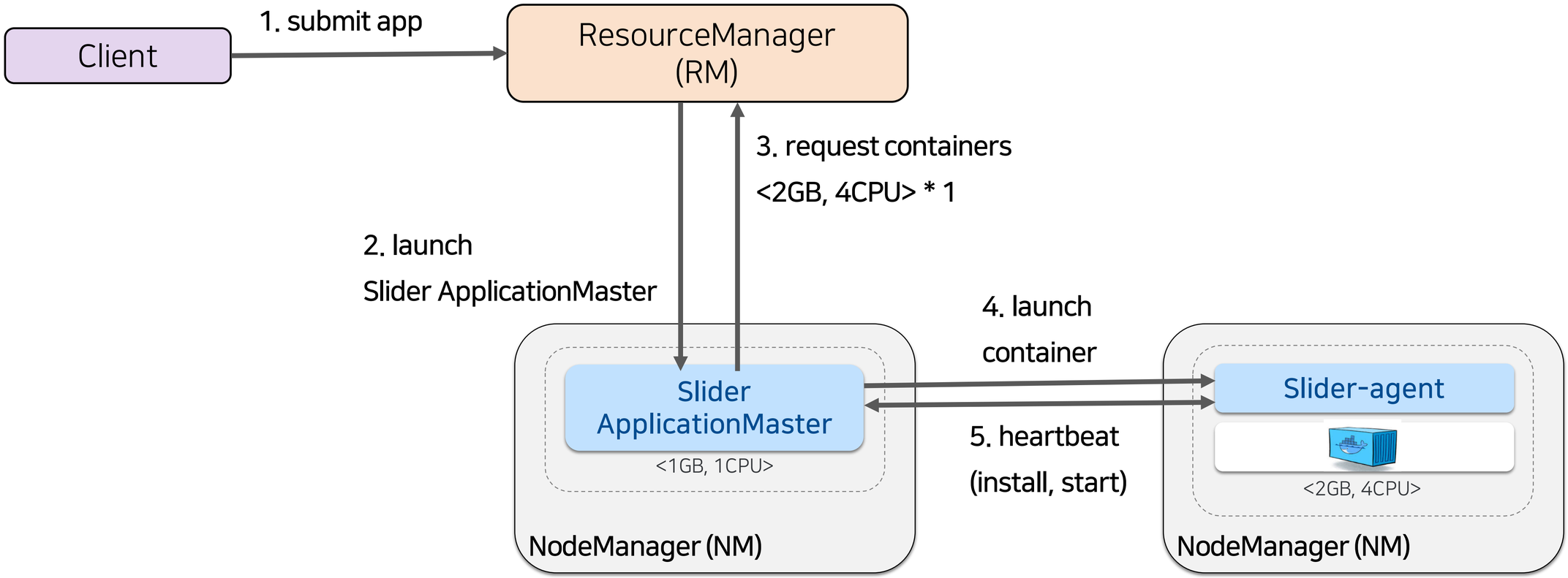
DEVIEW 2018

Docker on Slider

- Slider 를 이용해 *docker pull*, *docker run* 실행

```
import sys
import time
from resource_management import *

class Docker(Script):
    def install(self, env):
        import params
        env.set_params(params)

        cmd = format("/usr/bin/docker pull {docker_image}")
        Execute(cmd)

    def start(self, env):
        import params
        env.set_params(params)

        process_cmd = format("/usr/bin/docker run {docker_option}
        Execute(cmd)
```

2.6 Docker on Slider



3.
Considerations & Issues

DEVIEW
2018

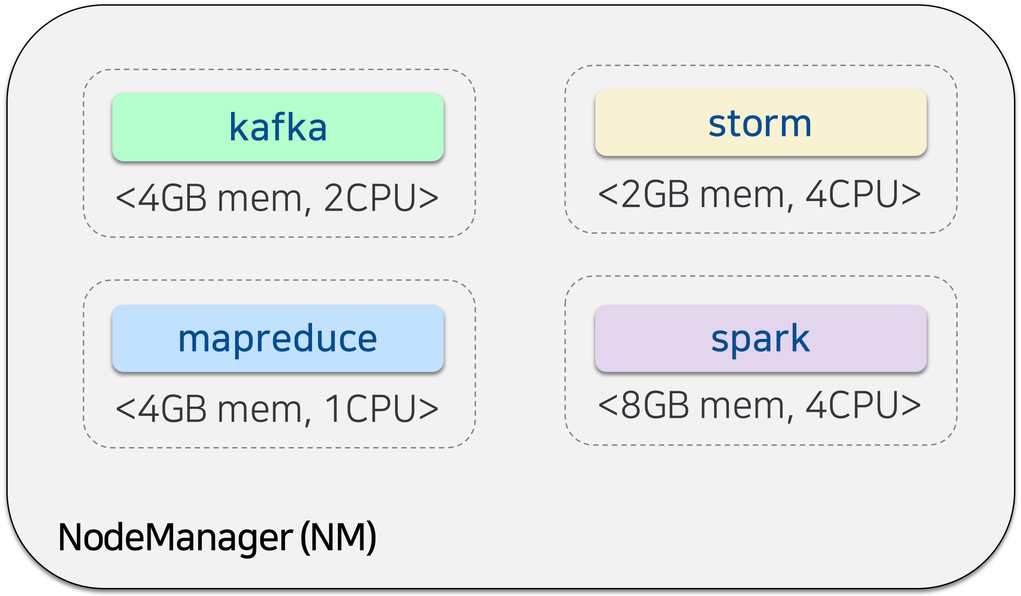
3.1 Resource Isolation

DEVIEW 2018

Considerations

- 다양한 앱이 동시에 실행

  CPU-intensive, memory-intensive …
- YARN 컨테이너에 할당된 리소스 보장
- YARN 컨테이너 사이의 간섭 문제 해결



3.1 Resource Isolation

CPU

- cgroup cpu.shares

Network – outbound bandwidth (YARN-2140)

- cgroup net_cls, linux tc tool

Disk I/O (YARN-2619)

- cgroup blkio

```
/cgroup/cpu
├── cgroup.event_control
├── cgroup.procs
├── cpu.cfs_period_us
├── cpu.cfs_quota_us
├── cpu.rt_period_us
├── cpu.rt_runtime_us
├── cpu.shares
├── cpu.stat
├── hadoop-yarn
|   ├── cgroup.event_control
|   ├── cgroup.procs
|   ├── container_e21_1536645454264_0003_01_000002
|   |   ├── cgroup.event_control
|   |   ├── cgroup.procs
|   |   ├── cpu.cfs_period_us
|   |   ├── cpu.cfs_quota_us
|   |   ├── cpu.rt_period_us
|   |   ├── cpu.rt_runtime_us
|   |   ├── cpu.shares
|   |   ├── cpu.stat
|   |   ├── notify_on_release
|   |   └── tasks
|   ├── container_e21_1536645454264_0003_01_000003
|   |   ├── cgroup.event_control
```

3.1 Resource Isolation - troubleshooting

DEVIEW 2018

[문제] Slider 로 Docker 를 실행하면 YARN 컨테이너 cgroup 으로 실행되지 않음

[해결] --cgroup-parent <YARN Container cgroup hierarchy>

```
/cgroup/cpu
├── cgroup.event_control
├── cgroup.procs
├── docker
│   ├── 80e44ff54da6c10bd7ac83069a54017d9d03ddc280
│   │   ├── cgroup.event_control
│   │   ├── cgroup.procs
│   │   ├── cpu.cfs_period_us
│   │   ├── cpu.cfs_quota_us
│   │   ├── cpu.rt_period_us
│   │   ├── cpu.rt_runtime_us
│   │   ├── cpu.shares
│   │   ├── cpu.stat
│   │   ├── notify_on_release
│   │   └── tasks
├── hadoop-yarn
│   ├── cgroup.event_control
│   ├── cgroup.procs
│   ├── container_e21_1536645454264_0006_01_000005
│   │   ├── cgroup.event_control
```

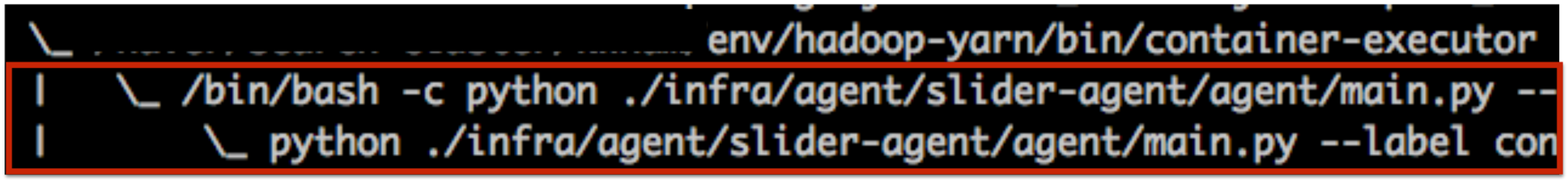
```
/cgroup/cpu
├── cgroup.event_control
├── cgroup.procs
├── cpu.cfs_period_us
├── cpu.cfs_quota_us
├── cpu.rt_period_us
├── cpu.rt_runtime_us
├── cpu.shares
├── cpu.stat
├── hadoop-yarn
│   ├── cgroup.event_control
│   ├── cgroup.procs
│   ├── container_e21_1536645454264_0005_01_000002
│   │   ├── 2f229ed9fcc1554e983f9f83563c904f68760d
│   │   ├── cgroup.event_control
│   │   ├── cgroup.procs
│   │   ├── cpu.cfs_period_us
│   │   ├── cpu.cfs_quota_us
│   │   ├── cpu.rt_period_us
│   │   ├── cpu.rt_runtime_us
```

3.1 Resource Isolation

DEVIEW 2018

Memory

- NM 의 ContainersMonitor 에서 사용 메모리 모니터링
- YARN 컨테이너 할당 메모리를 초과하는지 확인
- slider-agent 의 하위 프로세스 또는 그룹



3.1 Resource Isolation - troubleshooting

DEVIEW 2018

[문제] Slider 로 Docker 컨테이너를 실행하면 메모리가 감시되지 않음

- Docker 데몬의 하위 프로세스로 실행

```
/usr/bin/dockerd --exec-opt native.cgroupdriver=cgroupfs --live-restore
 \_ docker-containerd --config /var/run/docker/containerd/containerd.tom
 |   \_ docker-containerd-shim -namespace moby -workdir       /docker/co
 |       \_ python -m SimpleHTTPServer 8080
```

- slider-agent 의 하위 프로세스가 아님 (메모리 감시 제외)

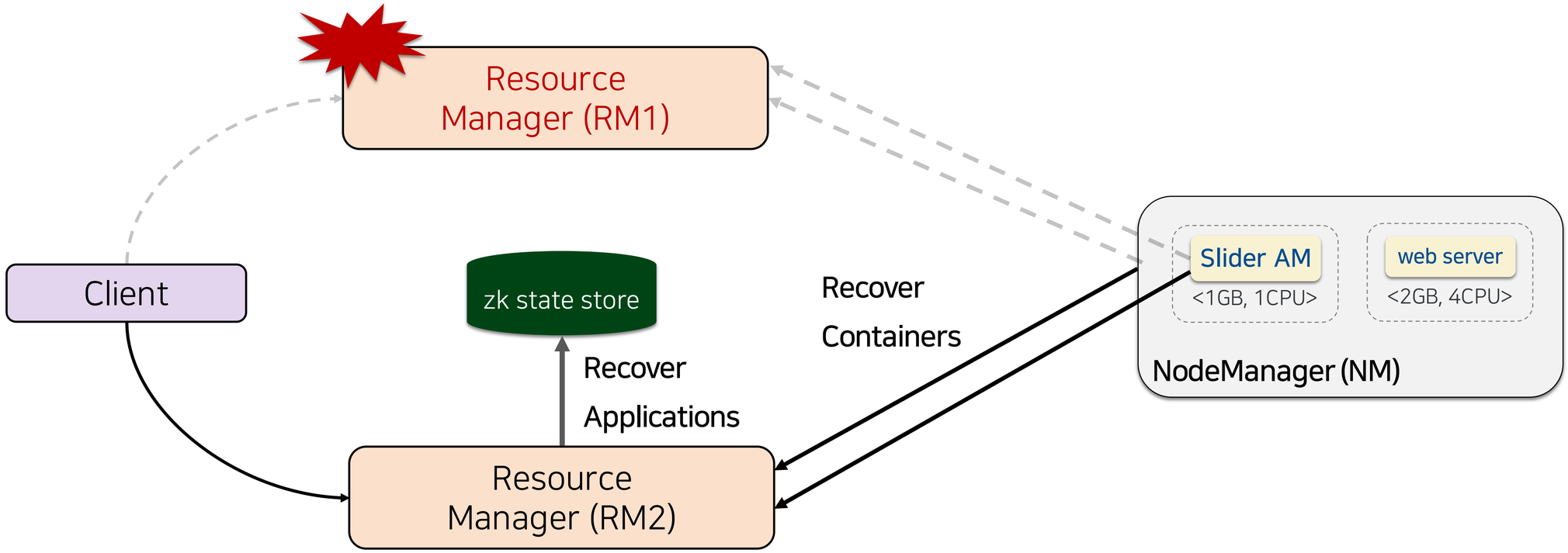
[해결] --memory <YARN Container 할당 메모리>

Reliability

3.2 Reliability of RM

DEVIEW 2018

RM HA with work-preserving recovery

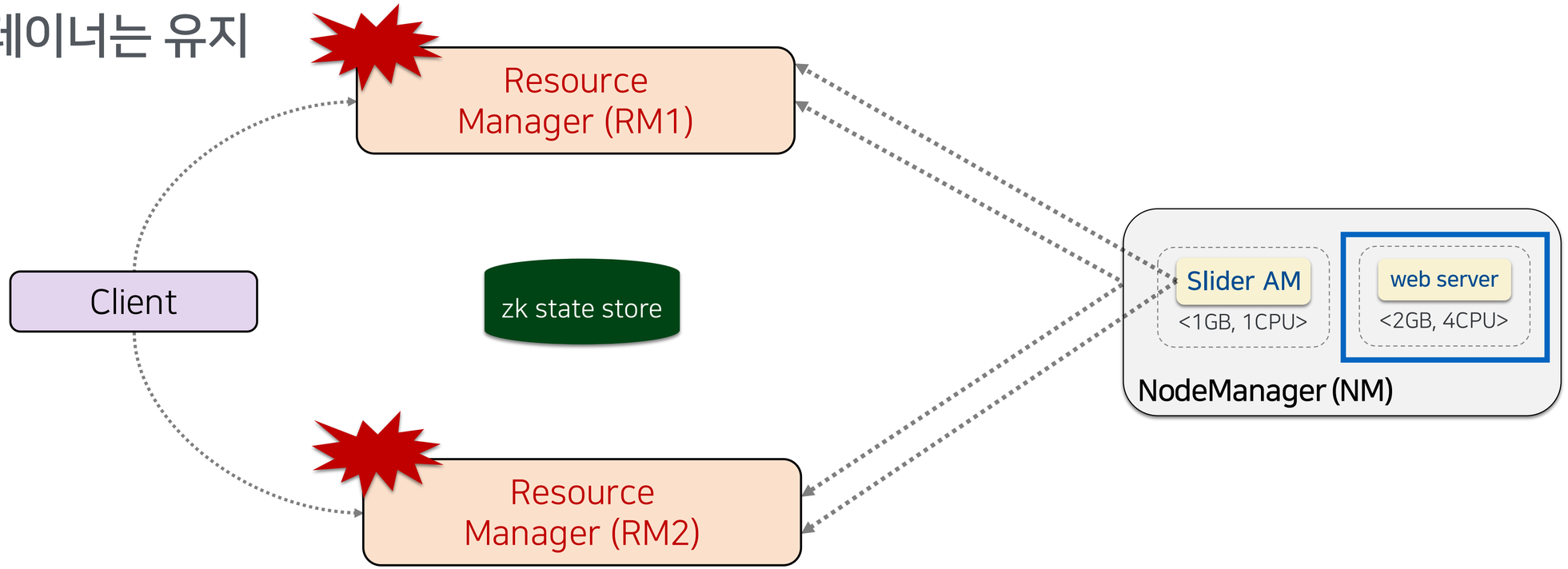


3.2 Reliability of RM

DEVIEW 2018

RM 이 모두 중단되면?

- Client 는 새로운 Application 을 요청하지 못함
- AM 은 새로운 컨테이너를 요청하지 못함
- **실행중인 컨테이너는 유지**



3.2 Reliability of RM

DEVIEW 2018

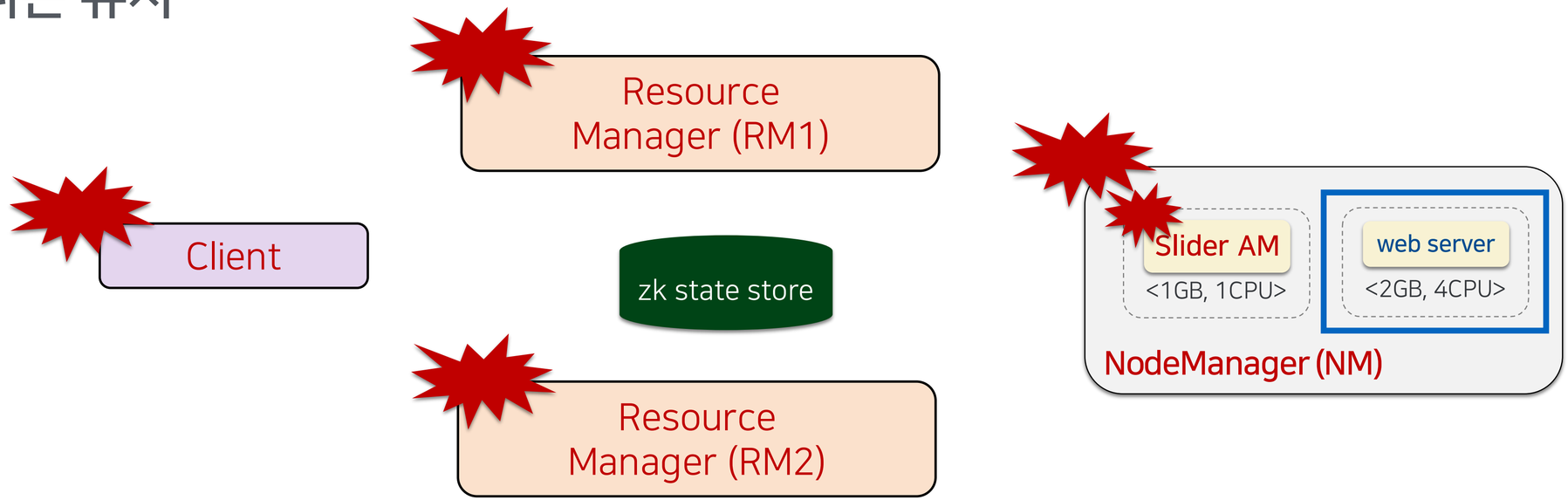
RM 이 계속 복구되지 못하면?

- Client, NM, AM 은 default 로 15분간 30초 간격으로 (30회) 재시도 후 종료

  yarn.resourcemanager.connect.max-wait.ms=900000

  yarn.resourcemanager.connect.retry-interval.ms=30000

- 실행중인 컨테이너는 유지



3.2 Reliability of RM - troubleshooting

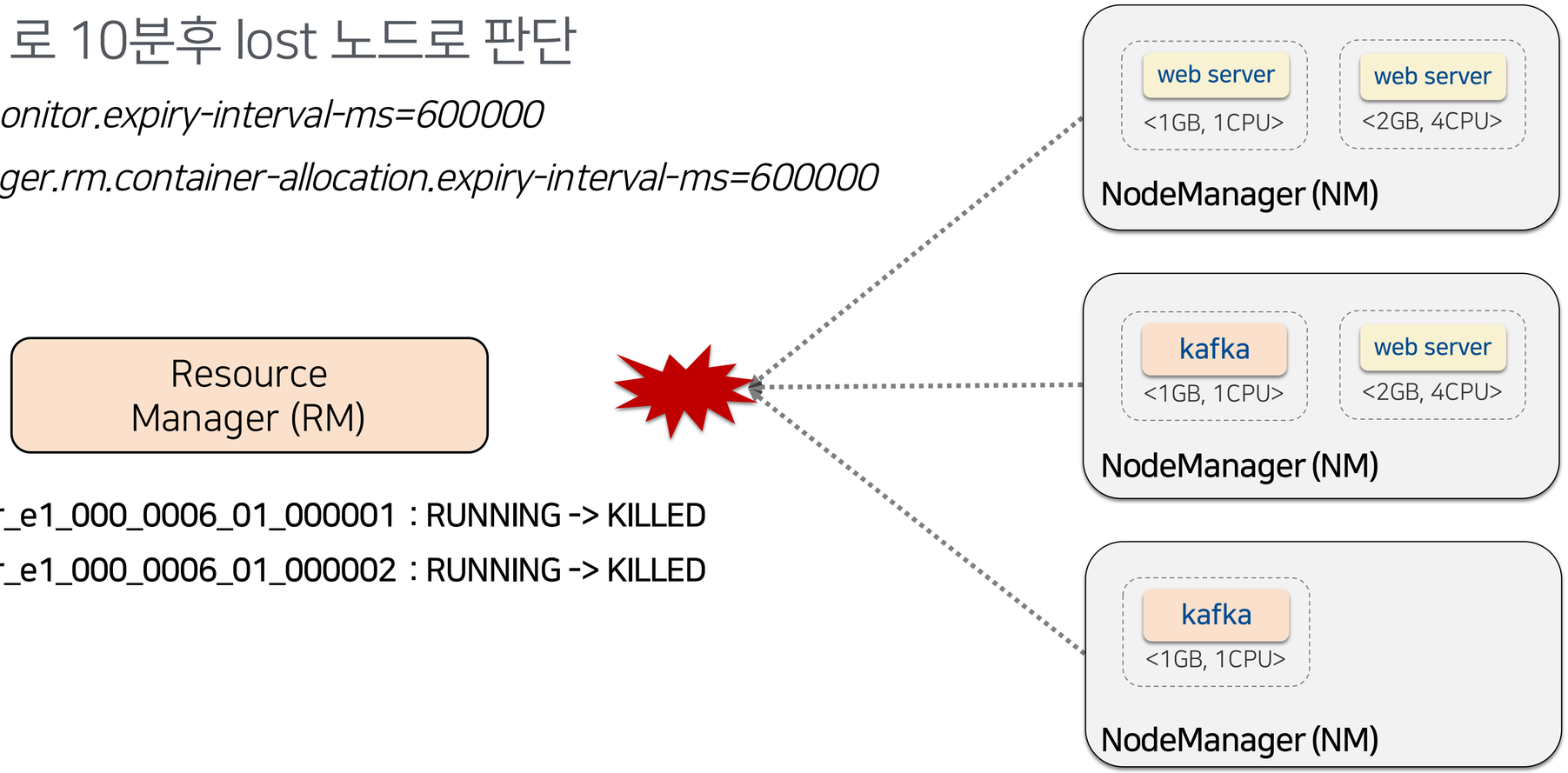
DEVIEW 2018

[문제] RM 과 NM 사이의 네트워크 장애

- RM 은 default 로 10분후 lost 노드로 판단

  yarn.nm.liveness-monitor.expiry-interval-ms=600000

  yarn.resourcemanager.rm.container-allocation.expiry-interval-ms=600000

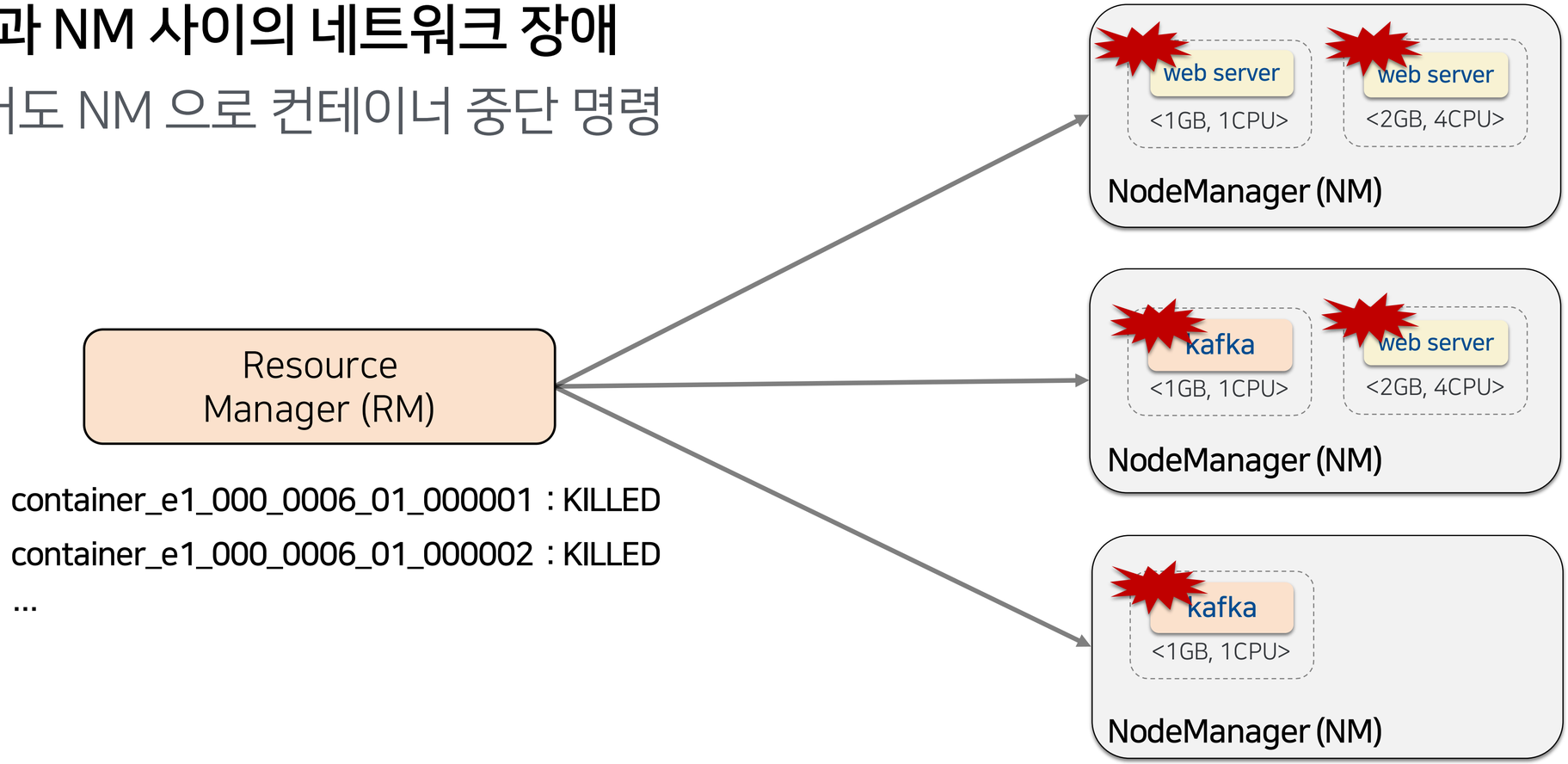


3.2 Reliability of RM - troubleshooting

DEVIEW 2018

[문제] RM 과 NM 사이의 네트워크 장애

- 복구되어도 NM 으로 컨테이너 중단 명령



3.2 Reliability of RM - troubleshooting

DEVIEW 2018

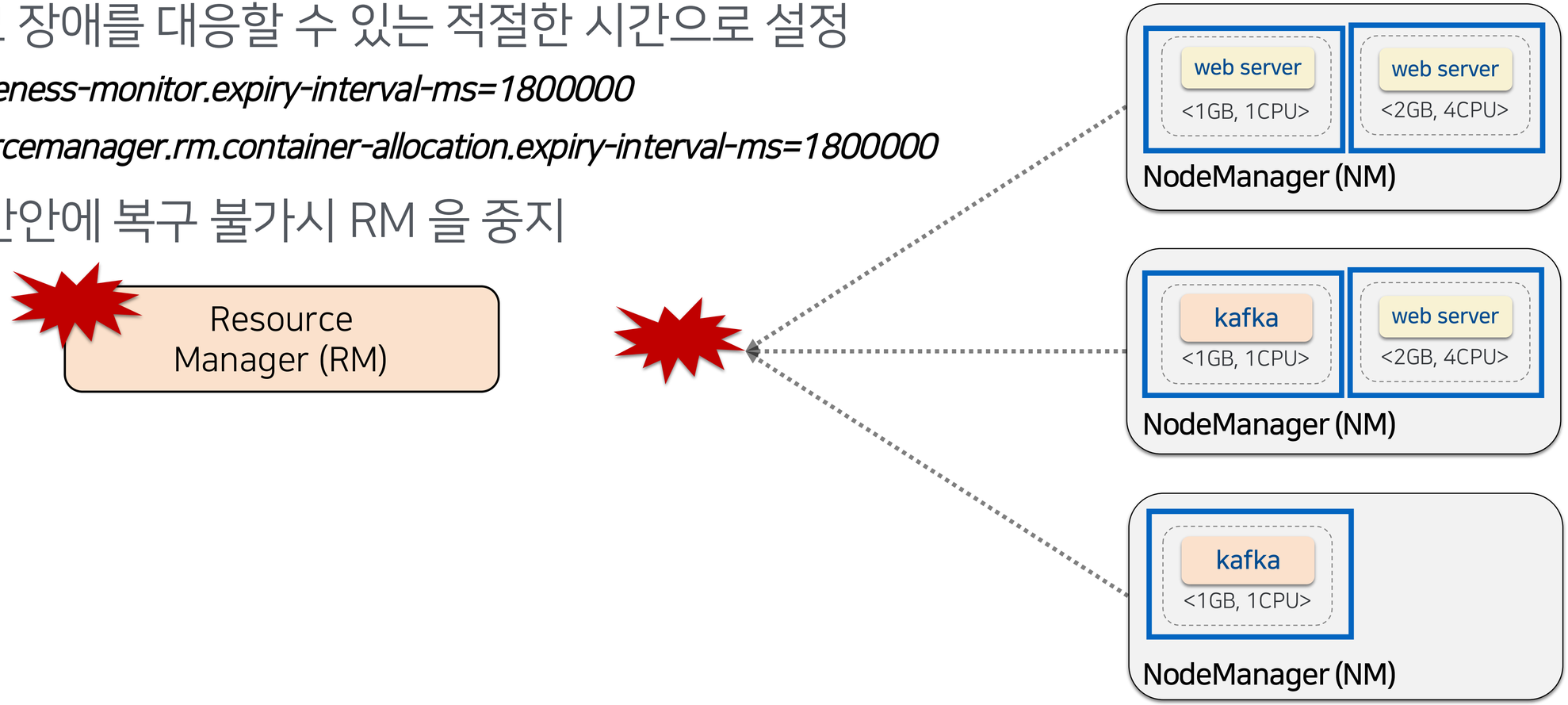
[해결] RM 과 NM 사이의 네트워크 장애 대비

- 네트워크 장애를 대응할 수 있는 적절한 시간으로 설정

  yarn.nm.liveness-monitor.expiry-interval-ms=1800000

  yarn.resourcemanager.rm.container-allocation.expiry-interval-ms=1800000

- 설정 시간안에 복구 불가시 RM 을 중지



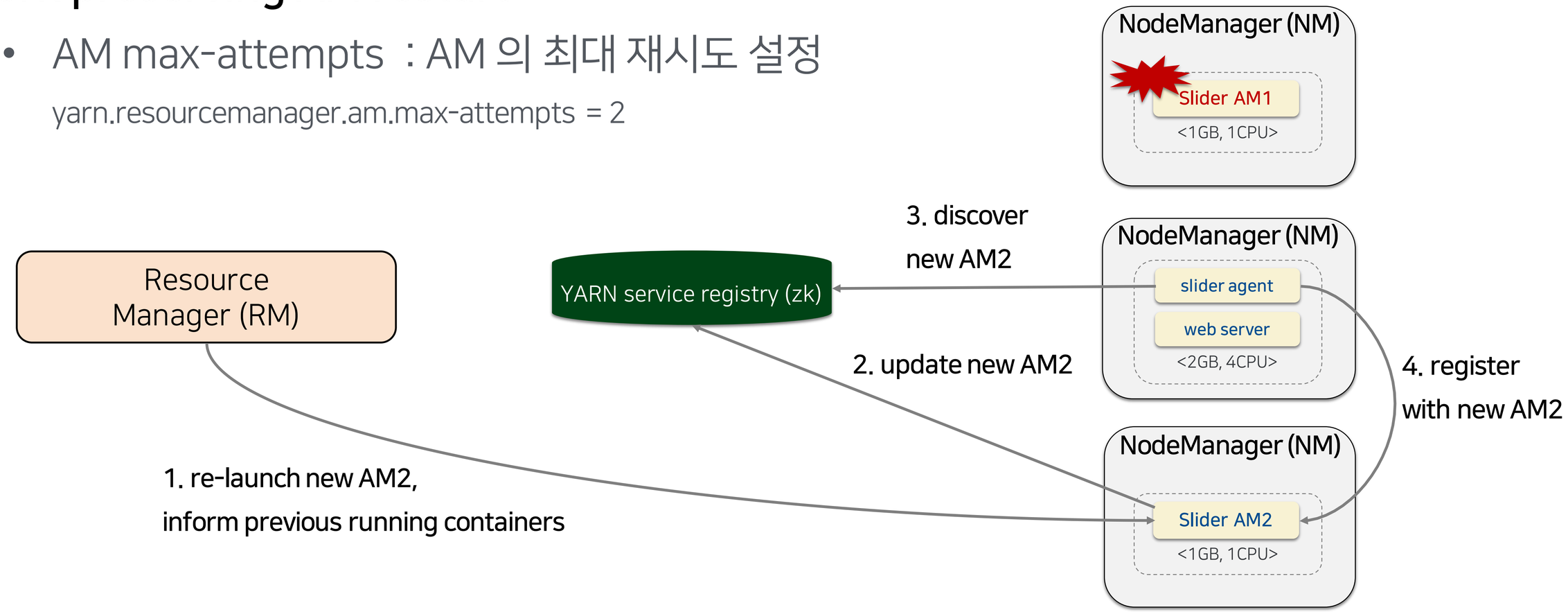
3.2 Reliability of AM

DEVIEW 2018

work preserving AM restart

- AM max-attempts : AM 의 최대 재시도 설정

  yarn.resourcemanager.am.max-attempts = 2



3.2 Reliability of AM

DEVIEW 2018

AM retry window

- longlived 앱의 경우 실패가 누적되어 AM max-attempts 초과
- AM retry-window : 일정 시간 동안의 실패 횟수만 고려

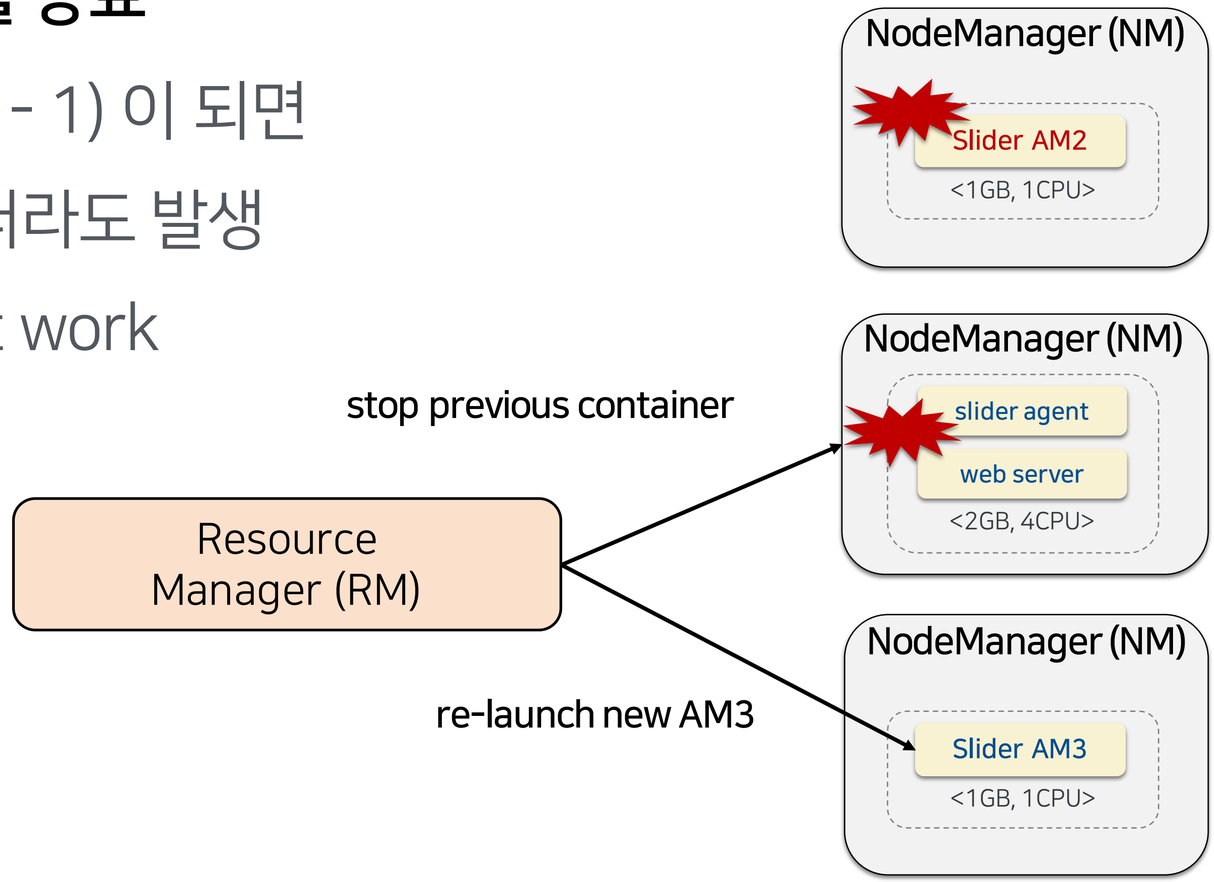
  yarn.resourcemanager.am.retry-count-window-ms = 300000 (slider 설정)

3.2 Reliability of AM - troubleshooting

DEVIEW 2018

[문제] AM 재시작 과정 중 RM 이 컨테이너를 종료

- 마지막 AM 재시작 (AM max-attempts - 1) 이 되면 AM retry-window 이후 AM 이 실패하더라도 발생
- YARN-6153 keepContainer does not work when AM retry window is set
- Fixed in 2.9.0, 3.0.0-alpha4



3.2 Reliability of AM - troubleshooting

DEVIEW 2018

[해결] AM 실패 대비하기

- AM max-attempts 설정 늘리기

  slider upgrade 로 인한 AM 재시작 고려

  yarn.resourcemanager.am.max-attempts = 5
- YARN-6153 패치 적용
- SLIDER-1253, SLIDER-1256

  Slider AM 실패시 실행중인 컨테이너를 종료시키는 문제 수정
- SLIDER-1221

  Slider AM 과 RM 간의 연결 실패시의 split brain 문제 해결

Scheduling

3.3 Scheduling

DEVIEW 2018

Considerations

- 중요한 서비스를 위한 전용 공간 확보 (Node Partition)
- 앱이 필요로 하는 리소스 할당 (Node Partition)
- Anti-affinity 와 같이 장애 대비를 고려한 컨테이너 할당 (Slider Placement Policy)

3.3 Scheduling

DEVIEW 2018

Node Partition

- 전용의 리소스 확보
- 실서비스 용도

default partition

NM NM ... NM

NM NM ... NM

추천서비스 개발자

recommend partition

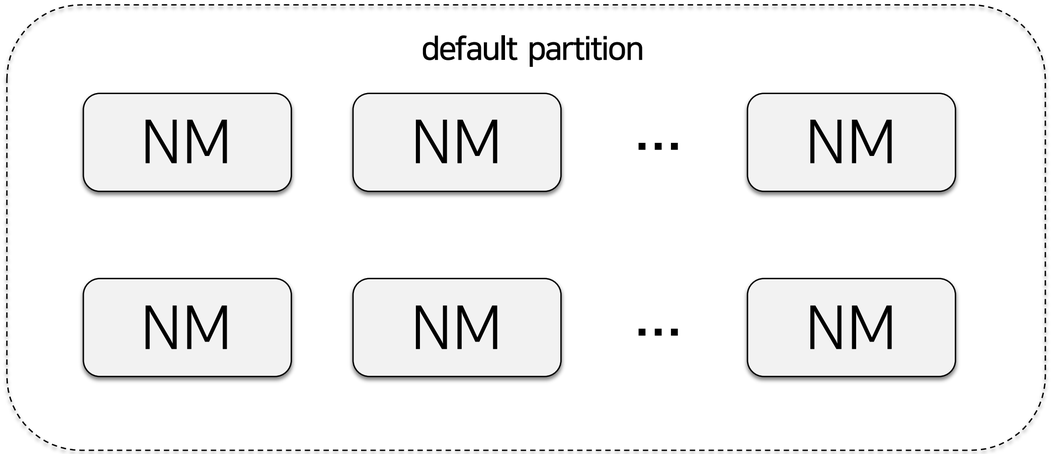
NM NM ... NM

3.3 Scheduling

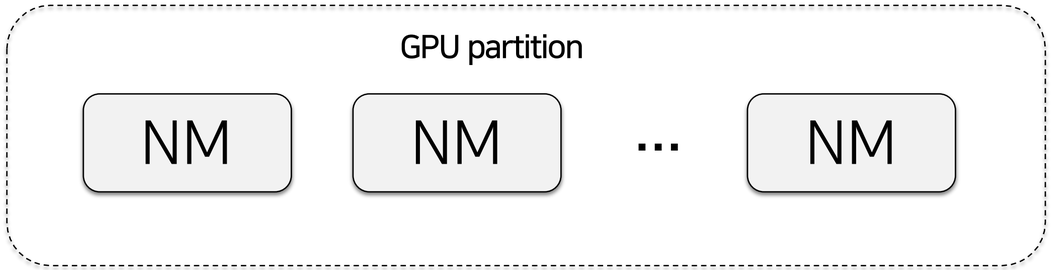
DEVIEW 2018

Node Partition

- 리소스 구분 (GPU, SSD, Large Disk…)
- 앱이 필요로 하는 리소스 할당



딥러닝 작업자



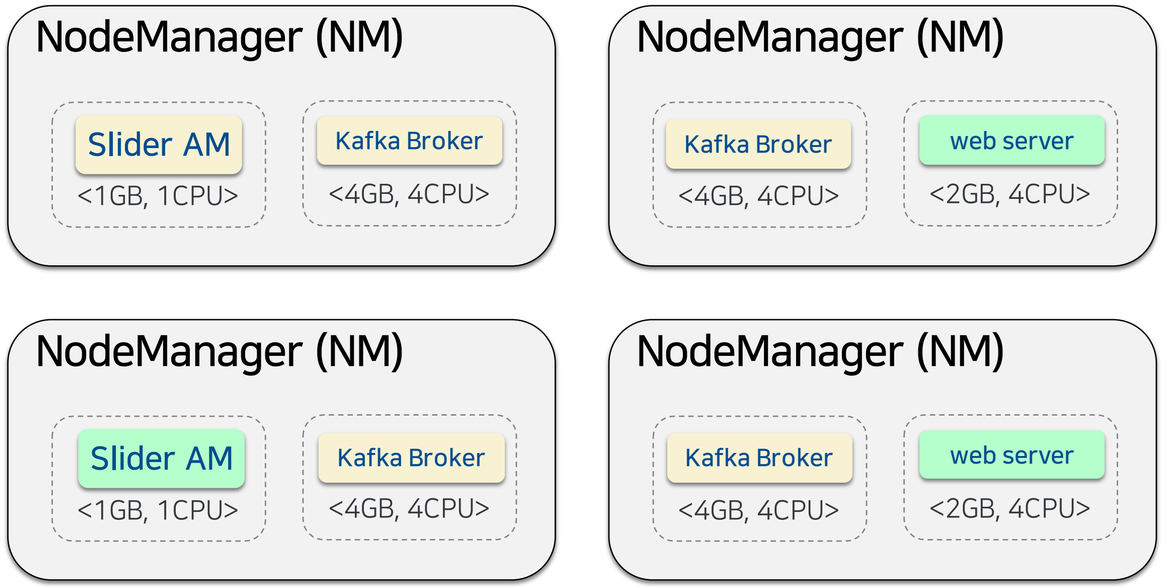
3.3 Scheduling

DEVIEW 2018

Slider Placement Policy

- default - 가능한 최근 실행되었던 노드
- static – 최근 실행되었던 노드에만
- anywhere – 어디든 상관없음
- **anti-affinity – 한 노드에 하나만 (장애 대비 가능)**

RoleHistory (HDFS)

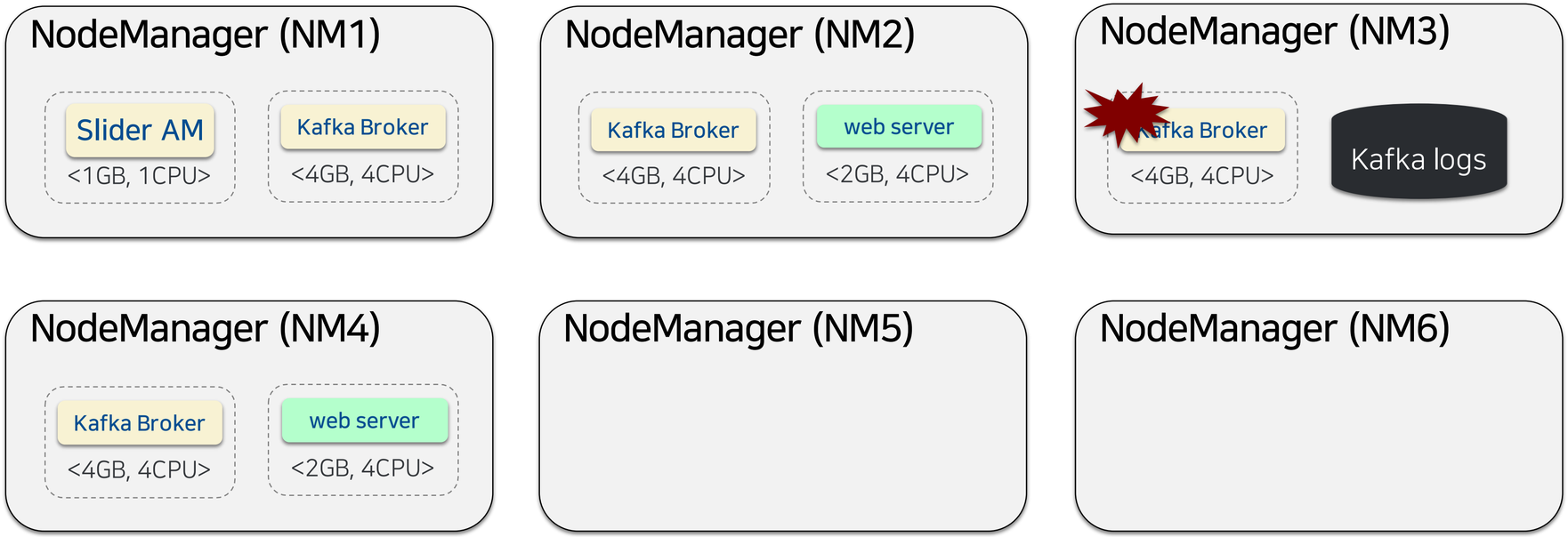


3.3 Scheduling - troubleshooting

DEVIEW 2018

[문제] 데이터를 로컬 디스크에 저장하는 앱의 경우 데이터 재사용이 어려움

- kafka, elasticsearch 는 데이터를 로컬 디스크에 저장
- anti-affinity 는 최근 실행되었던 노드를 고려하지 않음

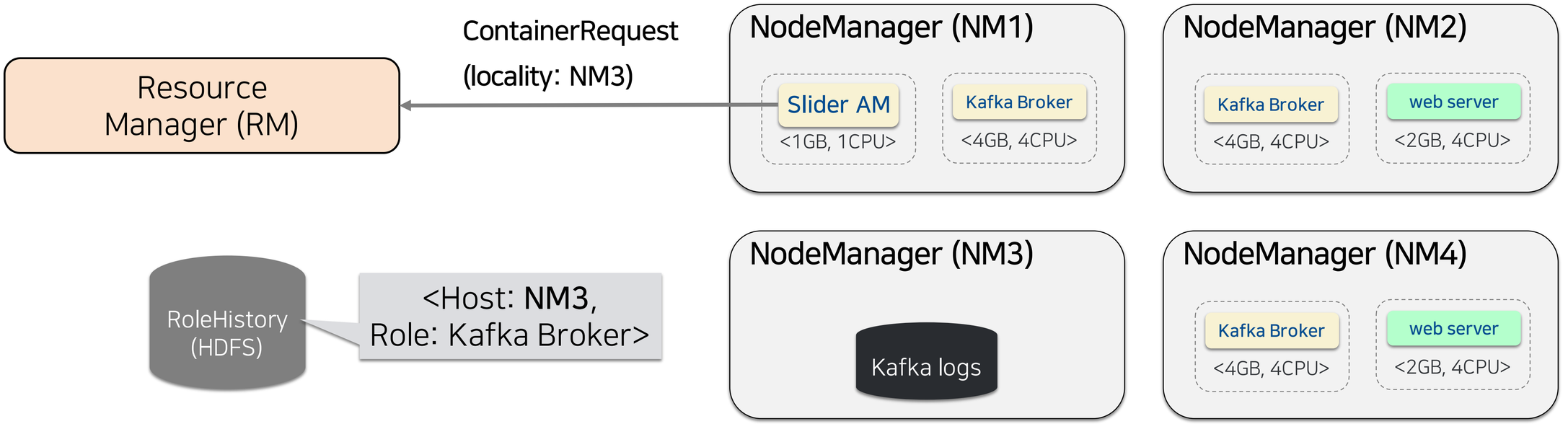


3.3 Scheduling - troubleshooting

DEVIEW 2018

[해결] anti-affinity placement policy 개선

- default + anti-affinity : 최근 실행되었던 노드를 우선하면서 한 노드에 하나씩
- SLIDER-1225 the Anti-Affinity placement policy should respect the role-history

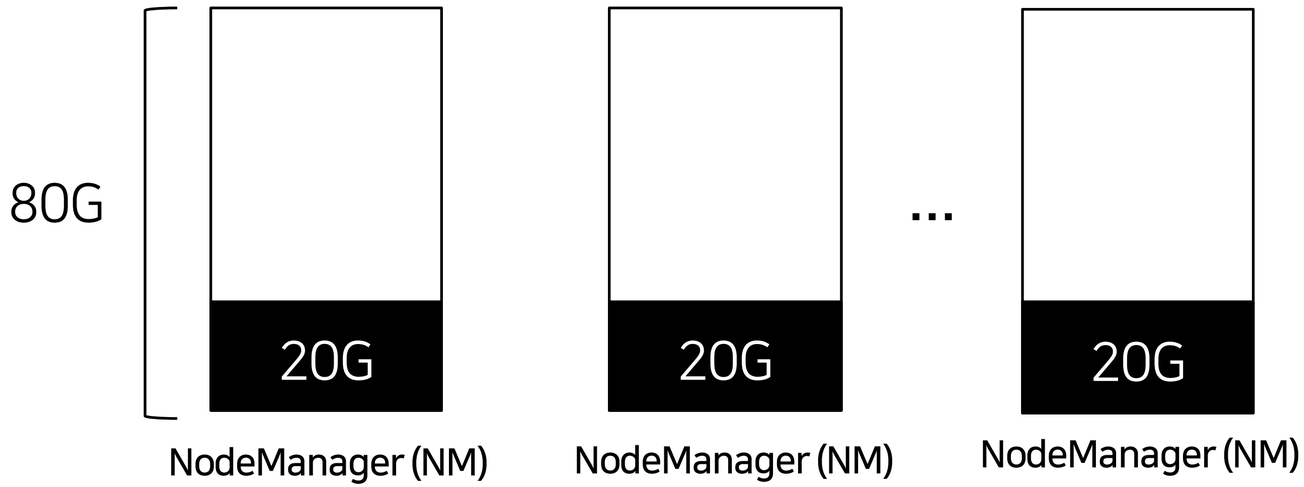


3.3 Scheduling - troubleshooting

DEVIEW 2018

[문제] Fragmentation

- 전체 리소스는 75% 여유
- 61G 컨테이너 요청은 할당 불가

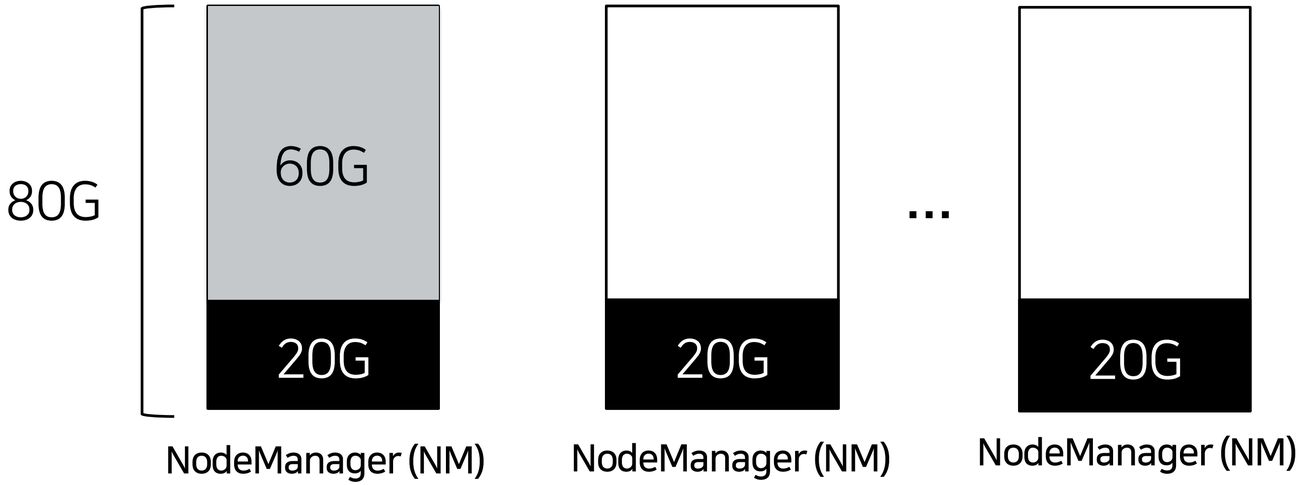


3.3 Scheduling - troubleshooting

DEVIEW 2018

[문제] Reservation

- YARN 스케줄러는 리소스가 부족한 노드를 Reservation
- Reservation 노드에서 사용중인 리소스 반환을 대기
- Reservation 리소스는 다른 유저/앱에 할당 불가

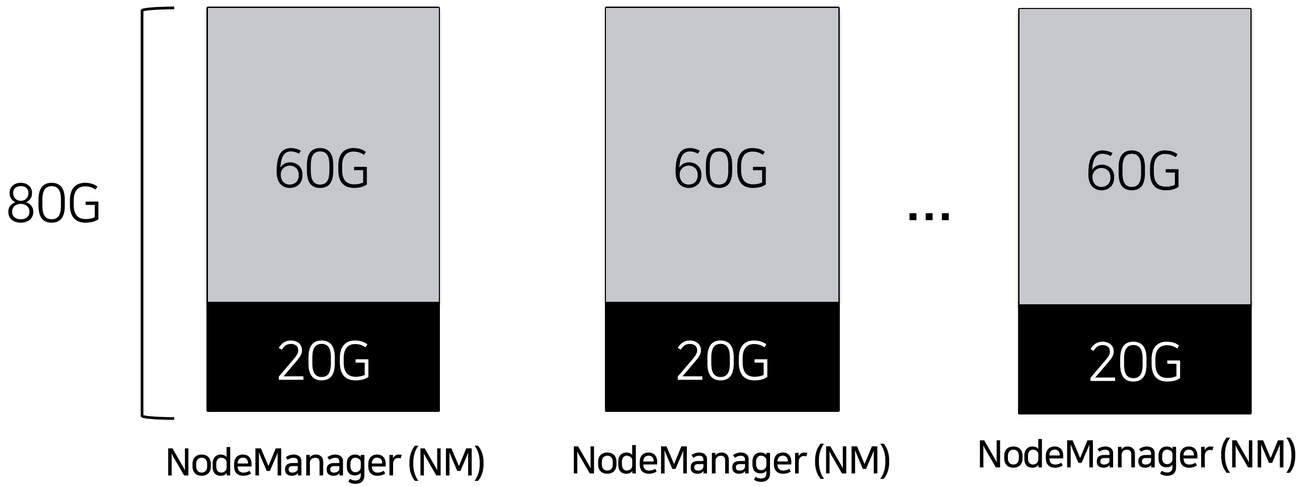


3.3 Scheduling - troubleshooting

DEVIEW 2018

[문제] Reservation

- YARN 스케줄러는 계속해서 할당이 안되면 Reservation 노드를 늘려나감
- Large 컨테이너 1개 요청으로 클러스터 전체 리소스를 점유



3.3 Scheduling - troubleshooting

DEVIEW 2018

[해결] Fragmentation / Reservation 문제 대비

- 컨테이너 최대 리소스는 가능한 작게 설정

  yarn.scheduler.maximum-allocation-vcores = 8

  yarn.scheduler.maximum-allocation-mb = 32768
- 큐별 한 사용자의 최대 사용량 제한 (reservation 제한)

  yarn.scheduler.capacity.<queue-path>.user-limit-factor = 0.1
- Scale up 보다는 Scale out (10G, 5개 보다는 5G, 10개로 요청)
- Queue Priorities (YARN-5864)

  큐별 우선순위 설정

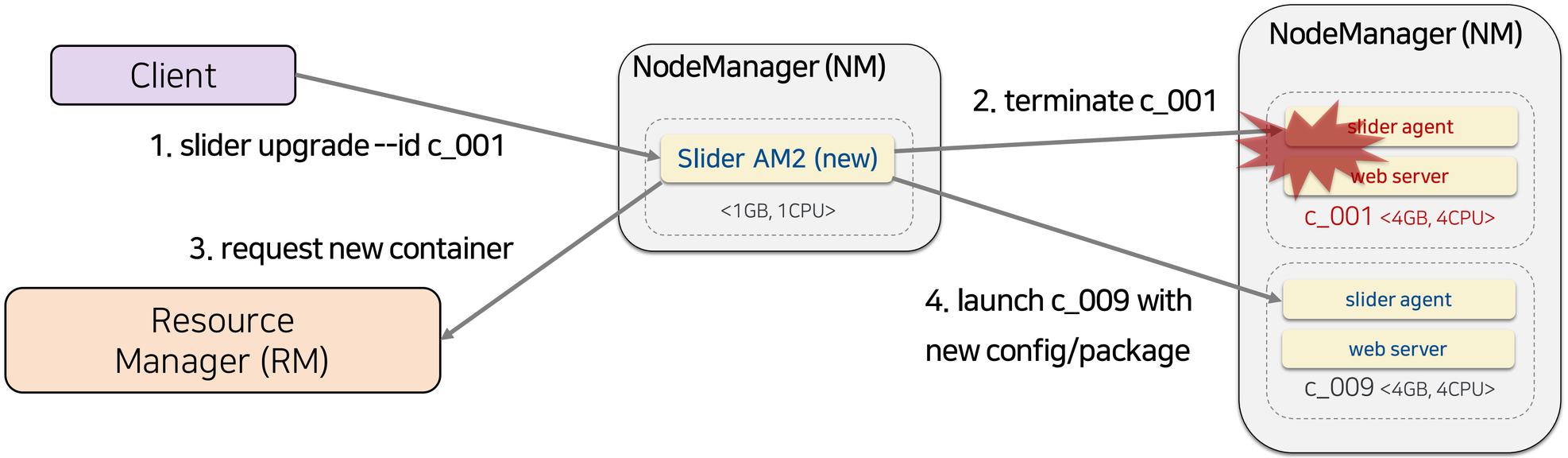
  낮은 우선순위 컨테이너를 Preemption 하여 할당

3.4 Upgrade / Reconfigure

DEVIEW 2018

Slider Rolling Upgrade

- 실행중인 앱의 패키지나 설정의 변경
- 새로운 설정으로 AM 재시작, 각 YARN 컨테이너 재시작 (반환 후 재할당)



3.4 Upgrade / Reconfigure - troubleshooting

DEVIEW 2018

[문제] YARN 컨테이너 재할당이 오래 걸림

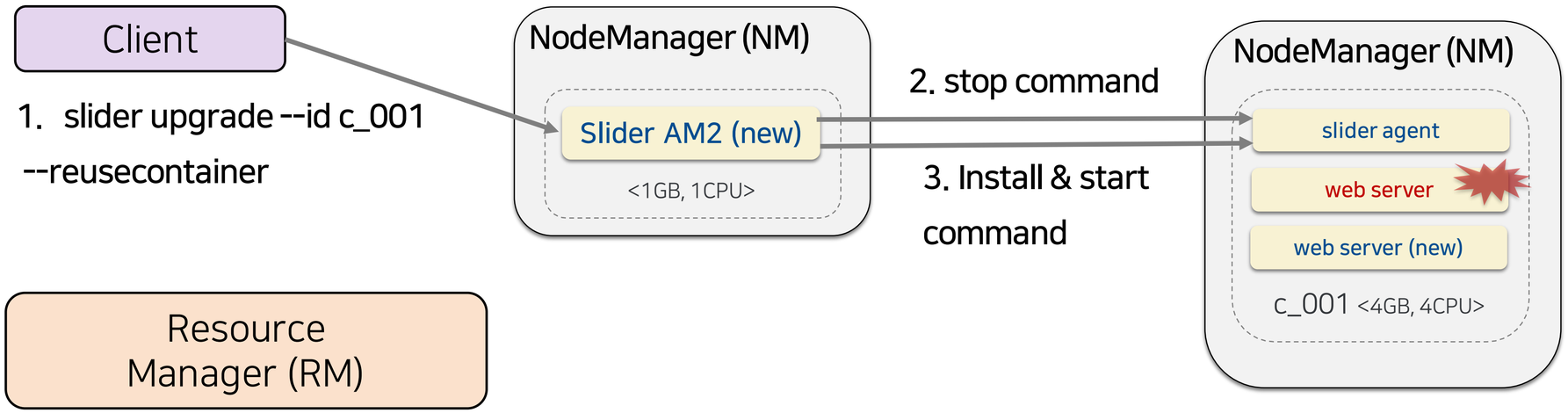
- 동시에 여러 사용자/앱에서의 리소스 요청
- 반환한 리소스의 재할당을 보장하기 어려움
- 재할당까지 오랜 시간이 소요될 수 있음

3.4 Upgrade / Reconfigure - troubleshooting

DEVIEW 2018

[해결] YARN 컨테이너를 재사용할 수 있도록 개선

- SLIDER-1265 Support upgrade with reusing YARN container
- 컨테이너 리소스 변경은 불가



3.5 Usability

DEVIEW 2018

Considerations

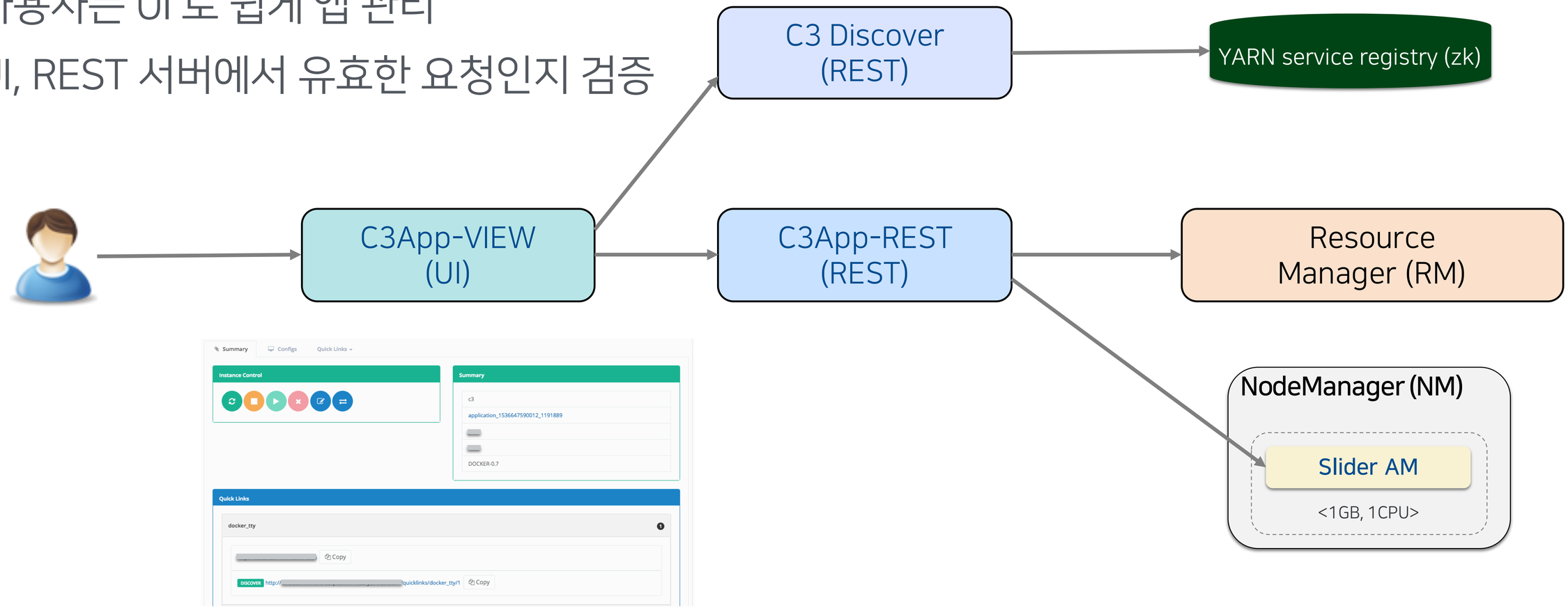
- Slider 는 CLI 만 지원
- Slider 패키지 설치 필요
- Slider 의 복잡한 설정을 이해하고 사용하기 어려움
- 잘못된 사용으로 실행중인 앱이 중단되는 경우

3.5 Usability

DEVIEW 2018

C3App

- 사용자는 UI 로 쉽게 앱 관리
- UI, REST 서버에서 유효한 요청인지 검증



4. Future work

DEVIEW 2018

Hadoop 3.1

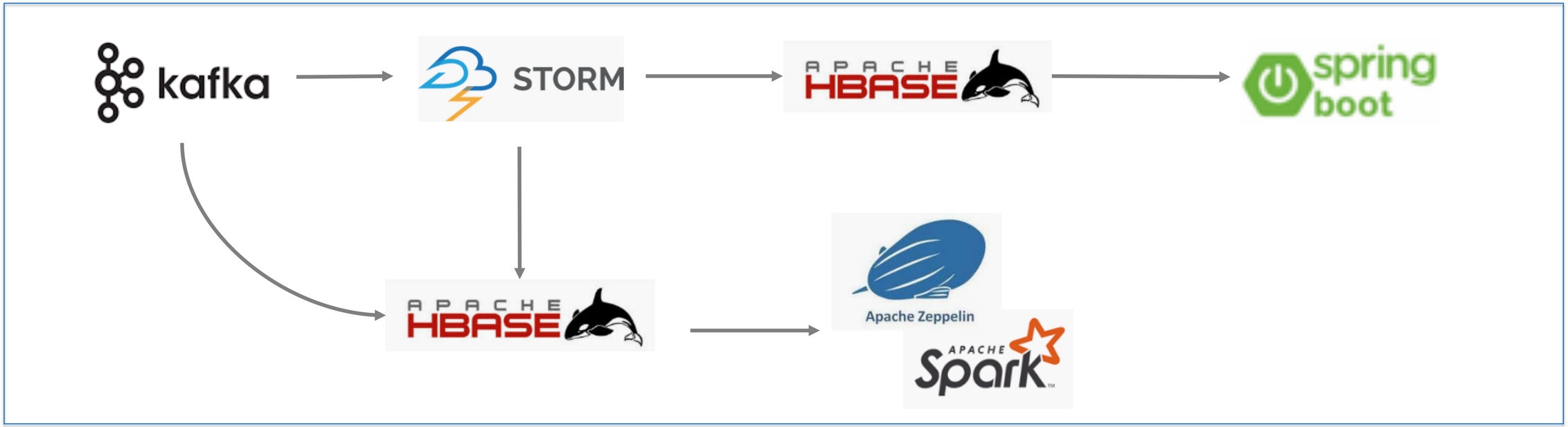
- Support Docker
- Yarn Service (Slider)
- Rich placement constraints

4. Future work

C3App Assembly

- 여러 앱들을 통합 관리
- service blueprint

my-service



Q & A

질문은 Slido에 남겨주세요.

sli.do

#deview

TRACK 1

Thank you