엣지 컴퓨팅 환경에서의 오류관리 기능 사례 분석 - StarlingX

김영선, 조재은, 김영한*
숭실대학교

zeroline@dcn.ssu.ac.kr, jayj@dcn.ssu.ac.kr, *younghak@ssu.ac.kr

A Case Study of Fault Management in Edge Computing System - StarlingX

Kim Young Sun, Cho Jae Eun, Kim Young Han*

Soongsil Univ.

요 약

기존 중앙집중화 되어 있던 클라우드 컴퓨팅은 모든 데이터를 중앙으로 모아서 처리하는 방식으로 데이터를 처리하는 과정에서 트래픽 문제, 과부하 문제, 데이터 지연 문제가 발생하게 된다. 이에 응답 시간을 개선하고 대역폭을 절약하기 위해서 필요한 곳에 연산과 데이터 스토리지를 도입하는 분산 컴퓨팅 패러다임인 엣지 컴퓨팅을 도입하고 있다. 하지만 여러 위치로 작업을 분산하기 때문에 각 엣지 클라우드에서 예기치 않은 동작이 발생했을 때 중앙 클라우드에서 중간에 수정 조치를 할 수 있도록 이벤트 및 알람으로 감지되어야 한다. 본 논문은 엣지 컴퓨팅 오픈소스 프로젝트인 StarlingX 의 오류관리 기능의 동작 방식, 모니터링 방법, 알람 정의 방식에 대해 분석하고자 한다.

Ⅰ. 서 론

전통적으로 클라우드 컴퓨팅은 대규모의 데이터 센터로 서비스가 중앙집중화되어 있다. 엣지 상의 IoT 장치와 데이터의 증가로 인해, 데이터 센터에서 감당하기에 네트워크 대역 요구사항이 한계치에 다다르게 되었다. 엣지 컴퓨팅은 응답 시간을 개선하고 대역폭을 절약하기 위해, 필요한 곳에 연산과 데이터 스토리지를 도입하는 분산 컴퓨팅이다. 엣지 컴퓨팅은 주로 네트워킹 요구 사항 또는 제약으로 인해 클라우드 컴퓨팅의 중앙집중식 접근 방식이 적절하지 않은 곳에 활용된다. 엣지 컴퓨팅을 이용하면 사용자에게 더욱더 빠르고 일관성 있는 환경을 제공할 수 있지만, 이로 인한 단점들도 존재한다. 데이터를 지리적으로 분산시켰기 때문에 상대적으로 각 사이트의 물리적 보안이 취약해질 수 있다. 악의적인 공격이나 네트워크의 단절 등으로 인한 문제 발생 위험률이 더 높아질 수 있다. 운영 중 이러한 예기치 않은 동작이 발생했을 때 일반적으로 중간에 수정 조치가 필요한 것들에 대한 알람이 필요하다. 오류관리 기능이 이러한 역할을 제공할 수 있다. 따라서 본 논문에서는 엣지 컴퓨팅 오픈 소스 프로젝트 중 하나인 StarlingX 에서 오류관리 기능으로 인프라에서 발생하는 다양한 동작을 어떻게 감지하고, 알람 또는 이벤트로 분류하여 알려주는지에 대해 분석하고자 한다.[1]

Ⅱ. StarlingX 의 오류관리 기능 분석

- 모니터링

인프라에서 발생하는 다양한 동작을 감지하고 어떤 알람 및 이벤트인지 분류하며 알려주는 동작을 하기 위해서는 다양한 요소를 고려하여 인프라의 상태를 모니터링 하여야 한다. StarlingX 에서 사용하는 모니터링을 위한 데몬 및 프로세스들은 표 1 과 같으며 각 프로세스들의 연결도는 그림 1 과 같다.[2] 표 1 의 위치에 해당하는 AC 와 BC 는 각각 Active Controller, Both Controller 를 의미한다. 아래의 모니터링 데몬을 통해 모니터링된 각 요소들은 오류관리 기능의 관리자(fmMgr)을 통해 SQL DB 에 저장되어 특정 임계 값 및 상태가 되면 알람 및 이벤트로 감지된다.

표 1. 모니터링 프로세스

프로세스	설명	위치
mtcAgent	Maintenance Agent 로 sysinv 와 통신하여 호스트 노드의 작동 및 가용성 상태를 관리.	AC
hwmond	Hardware Monitoring Daemon 으로 호스트 센서를 주기적으로 모니터링하고 허용 오차 범위를 벗어날 시 상태 보고.	AC
hbsAgent	Heartbeat Service Agent 로 활성화된 모든 호스트가 존재하는지, 네트워크에 응답하는지 확인.	BC
hostwd	HostWatchdog 으로 중요한 프로세스 쿼럼(hbsClient, mtcClient, SM, fsmond)을 모니터링하고 데이터 수집을 하는 하드웨어 및 소프트웨어 watchdog	ALL
pmond	Process Monitoring Daemon 으로 프로세스들을 모니터링하고 정상 작동하지 않을 시 재시작.	ALL
fsmond	Filesystem Monitoring Daemon 으로 파일 시스템 잠금을 감지.	ALL
hbsClient	hbsAgent 의 응답기.	ALL

mtcClient	mtcAgent 의 커맨드를 처리하는 핸들러.	ALL
lmond	Link Monitoring Daemon 으로 네트워크 링크 이벤트를 모니터링하고 해당 이벤트를 collectd 에 보고.	ALL
Collectd plugins	메모리, CPU, 인터페이스, 파일 시스템을 모니터링하고 임계 값을 넘을 시 mtcAgent 를 통해 호스트 상태를 degrade 로 트리거.	ALL

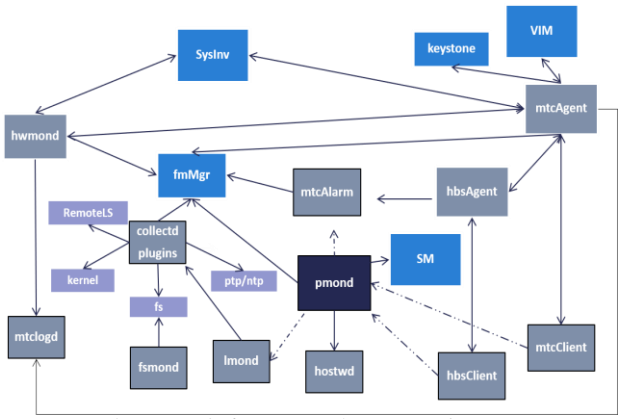

그림 1. 모니터링 프로세스들 간의 연결도

● 알람 정의

모니터링된 자원 및 호스트 상태를 기반으로 어떤 임계 값 및 상태에서 알람을 발생시키는지에 대해 정의하여야 한다. StarlingX 에서는 그림 2와 같이 /etc/fm/events.yml 에 알람을 정의하고 있다. <Alarm Group ID>.<Alarm Event ID>라는 알람 ID 를 주어 특정 알람 조건을 식별한다. Alarm Group ID 의 종류는 다음과 같다.

- 100 : 모니터링된 리소스
- 200 : 베어메탈
- 300 : 네트워크
- 400 : 고가용성(High Availability)
- 500 : 보안

또한 해당 알람에 대한 설명을 의미하는 Description, 발생한 알람의 타입과 인스턴스 정보를 의미하는 Entity_Instance_ID, 해당 알람이 일어난 host 의 상태를 무엇으로 변경하는지에 대한 Maintenance_Action 과 같은 필드도 정의한다. 기존에 존재하는 알람 외에도 /etc/fm/events.yml 파일에 원하는 알람 및 이벤트에 대해 그림 2와 같이 정의하여 감지할 수 있다.

```
#-------------------------------------------------------------------------------
#   Monitored Resource Alarms
#-------------------------------------------------------------------------------


100.101:
    Type: Alarm
    Description: |-
        Platform CPU threshold exceeded; threshold x%, actual y% .
              CRITICAL @ 95%
              MAJOR    @ 90%
              MINOR    @ 80%
    Entity_Instance_ID: host=<hostname>
    Severity: [critical, major, minor]
    Proposed_Repair_Action: "Monitor and if condition persists, contact next lev
el of support."
    Maintenance_Action:
        critical: degrade
        major: degrade
    Inhibit_Alarms:
    Alarm_Type: operational-violation
    Probable_Cause: threshold-crossed
    Service_Affecting: False
    Suppression: True
    Management_Affecting_Severity: major
    Degrade_Affecting_Severity: critical
```

그림 2. yaml 파일을 이용한 알람 정의

● 오류관리 기능 구조

StarlingX 의 오류관리 기능은 알람 및 이벤트 데이터를 발생시키고 유지하기 위한 인프라 서비스 프레임워크이다. 알람 및 이벤트 정의를 기반으로 그림 3의 구조를 가지며 그림 4와 같이 동작한다. StarlingX 여러 서비스들이 알람에 대한 데이터를 요청하면 FM Client 가 C API 와 Python API 를 사용할 수 있게 해주는 컴포넌트로써 REST API 인 FM API 를 이용해서 FM Manager 가 SQL DB 인 InfluxDB 에 접근하게 한다. FM Manager 가 InfluxDB 에 접근하여 활성화 되어 있는 알람 및 이벤트에 대해 데이터를 제공한다. 각 알람은 fm 명령어를 이용하여 CLI 로, 대시보드인 GUI 로 확인할 수 있으며 원하지 않는 알람 및 이벤트에 대해서 억제할 수 있다.

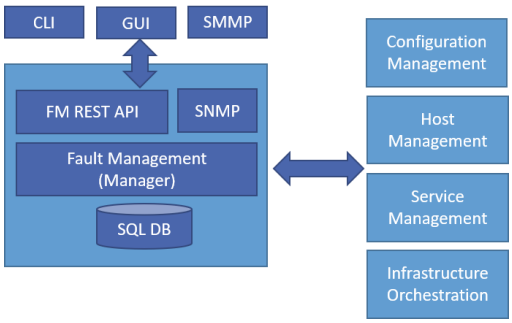

그림 3. 오류관리 기능 구조

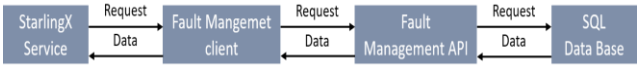

그림 4. 오류관리 기능의 동작 절차

Ⅲ. 결론

본 논문에서는 오류관리 기능의 필요성에 관해 설명하고, 실제로 오픈 소스에서 어떻게 사용되는지 StarlingX 의 사례를 통하여 분석하였다. 하지만 StarlingX 의 사례에서도 알 수 있듯이, 대부분의 인프라 오류관리 기능의 모니터링된 정보 및 알람을 기반으로 시스템 운영자가 수정 및 복원할 수 있지만 자동으로 복구 및 수정하는 기능이 부족하다. StarlingX 의 경우 엣지 사이트를 중앙에서 관리할 수 있도록 인프라를 구축하였지만, 엣지 사이트가 많아질수록 중앙의 한정된 인원으로 모든 엣지 사이트를 관리한다는 것은 비효율적이고, 문제를 파악하고 순차적으로 해결하는 데 시간이 걸릴 수 있다. 따라서 자동 복구 기능을 도입하여 간단한 작업이라도 자동으로 해결될 수 있도록 하는 것이 필요할 것이다.

ACKNOWLEDGMENT

"본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학 ICT 연구센터지원 사업의 연구결과로 수행되었음" (IITP-2020-2017-0-01633)

참 고 문 헌

[1] StarlingX : https://www.starlingx.io

[2] StarlingX Maintenance :
https://wiki.openstack.org/wiki/StarlingX