오픈스택 환경에서의 실시간 오픈소스 모니터링 시스템 비교 분석*

신승원[O1] 권기남[1] 고영훈[1] 남윤수[2] 박현찬[1]
[1] 전북대학교 컴퓨터공학과
[2] 전북대학교 소프트웨어공학과
tlstmdck@gmail.com,{ginami0129, dmsgh423, nys6635, hyunchan.park}@jbnu.ac.kr

Comparative analysis of real-time open-source monitoring system in OpenStack environment

Seoungwon Shin[O1] Ginam kwon[1] Younghun Go[1] Younsu Nam[2] Hyunchan Park[1]
[1] Division of Computer Science and Engineering, Jeonbuk National University
[2] Division of Software Engineering, Jeonbuk National University

요 약

사용자에게 가상머신을 제공하여 가상화된 자원들을 사용하는 오픈스택은 여러 서비스로 구성되어 있다 그 중에서도 모니터링 서비스는 다양한 서버 자원의 사용률 및 사용자의 인스턴스 상태를 관측하고 문제 발생시에 관리자에게 알림을 보내는 등 오픈스택을 운영하는데 있어 매우 중요한 역할을 수행한다. 본 논문에서는 오픈스택에서 제공하는 모니터링 서비스인 모나스카와 오픈스택 프로젝트가 아닌 모니터링 시스템인 프로메테우스를 비교 분석한다.

1. 서 론

클라우드 서비스를 구축하는데 필요한 소프트웨어를 제공하는 오픈소스 플랫폼인 오픈스택은 사용자에게 가상머신을 제공하여 가상화된 자원들을 사용하는 것을 목표로 한다. 이러한 오픈스택은 여러 프로젝트로 구성되어있는데 컴퓨팅 자원을 제공하는 nova, 네트워크 자원을 제공하는 neutron, 스토리지 자원을 제공하는 cinder 등이 있다. 그 외에 keystone, horizon 등 인증과 대시보드와 같은 필수적인 서비스를 제공하는 프로젝트도 함께 제공한다 [1].

추가적으로 반드시 필요한 기능으로 모니터링 기능이 있다. 서버의 cpu, memory 등 다양한 자원뿐만 아니라 서비스 프로세스의 상태, 사용자의 인스턴스의 사용량과 이상행위까지 관측해주는 서비스가 있어야 복잡한 클라우드 시스템의 상황을 쉽게 관찰할 수 있다. 오픈스택에서는 모니터링을 위해 모나스카 프로젝트를 제공한다. 그 외에도 오픈스택의 프로젝트는 아니지만, Prometheus, Zabbix, Nagios 등 다양한 모니터링 시스템들이 있다.

본 연구진은 이러한 모니터링 시스템들 중, 모나스카와 프로메테우스를 설치난이도, 데이터수집범위, 대시보드, 멀티테넌트(multi-tenant)의 4가지 요건에 대해 비교, 분석한다. 위 4가지 요건은 모니터링 시스템을 실제 활용하는데 있어 가장 중요하다고 판단되는 요건들을 제시한 것이다. 그 결과, 모나스카가 설치 난이도가 높지만, 오픈스택 환경에서 관리하기에 충분한 기능을 제공하고 있고, 프로메테우스는 기능이 다소 부족하지만, 설치 편의성이 높고, 관련 정보를 얻기가 매우 편리하다는 점을 파악하였다.

2. 모니터링 시스템 소개

2.1. 프로메테우스

프로메테우스는 2012년 Sound Cloud에서 개발한 메트릭 기반의 오픈소스 모니터링 시스템이고 2016년 CNCF(Cloud Native Computing Foundation)에 합류하였다. 프로메테우스는 풀링(pulling)방식으로 Node-Exporter 에이전트를 통하여 메인 서버에서 타겟들의 데이터를 수집하는 방식을 사용한다. 이를 통하여 Auto Scaling이 자주 발생하는 환경에서 대상의 정보가 변경되거나 추가되어도 Endpoint에 주기적으로 요청을 전송함으로써 데이터를 가져올 수 있다. 이러한 Endpoint 정보들은 Service discovery에 저장된다. Service discovery는

* 이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2020R1F1A1067365).

Endpoint의 정보를 관리해주고 데이터 수집 주기 와 같은 옵션을 정해주는 역할을 한다. Service discovery를 통해 수집된 메트릭은 텍스트로 변환 후에 시계열DB인 프로메테우스 TSDB(time series database)에서 Key-Value 형태로 타임스탬프와 함께 저장한 후 자체적인 Web-UI, Grafana [2]와 같은 다양한 대시보드로 이를 시각화 한다 [3].

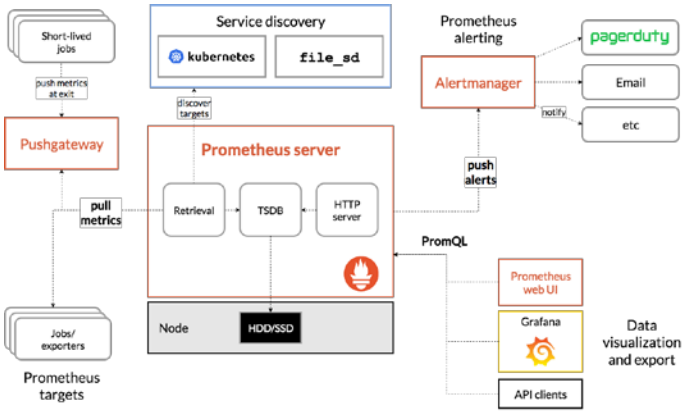

그림 1. 프로메테우스의 구조 [4]

2.2. 모나스카

모나스카는 오픈스택의 자체적인 프로젝트로 멀티테넌트, 높은 확장 가능성, 뛰어난 성능과 내결함성을 가진 REST API 기반의 오픈소스 모니터링 시스템이다. 모나스카는 CLI 및 Horizon에서 정의한 알림과 Agent가 수집하는 메트릭을 REST API인 Monasca API를 통해 실시간 비동기 처리를 하는 분산 메시지 큐인 Kafka로 이동을 한 후 알림 및 메트릭은 Persister를 통하여 시계열 DB인 InfluxDB로 저장이 되고 설정된 알림에 관하여 메트릭의 임계값을 초과하는 경우에는 Threshold를 통하여 메시지 큐에 등록을 하며 Notification을 통하여 저장된 메일로 관리자에게 알림을 전송한다. 이러한 동작은 분산되어서 병렬로 처리하기 때문에 매우 빠른 처리속도를 지닌 구조이다 [5].

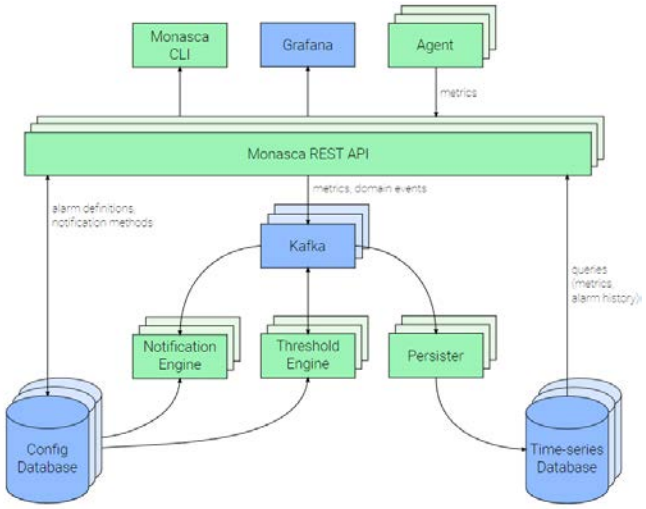

그림 2. 모나스카의 구조 [5]

3. 비교 분석

	프로메테우스	모나스카
설치난이도	쉬움	매우 어려움
데이터 수집 범위	exporter가 설치된 client (서비스, 가상머신에 별도 설치 필요)	오픈스택 서비스, 가상머신 (추가 작업 없음)
대시보드	Prometheus web을 이용	horizon에 통합
멀티테넌트	X	O

표 1. 모니터링 시스템 비교

첫째, 설치 면에서 프로메테우스는 이미 구현된 Container Image를 활용하여 컨테이너 관리 시스템인 Docker를 통해 HTTP서버와 자체적인 DB를 구축한다. 추가로 에이전트인 exporter나 대시보드인 Grafana를 설치하면 운영하는데 문제가 없다.
반면 모나스카의 경우, 메트릭, 알림 정보를 저장하는 InfluxDB, 요청을 처리하는 API, 메트릭을 수집하는 Agent뿐만 아니라 Persister, Thresholds, Notification 등 여러 엔진을 설치하여야 하고, 이러한 엔진이 활성화할 수 있게 Kafka, Zookeeper, Apache Storm을 설치하여야 한다. 이러한 복잡한 설치는 시간도 매우 많이 소요되고 설치 중 각 요소에 버전이 서로 일치하지 않아 오류가 발생할 확률도 높다.
둘째, 데이터의 수집 범위 면에서 프로메테우스의 경우에는 에이전트인 Exporter가 심어져 있는 Client를 측정하게 된다. 그리하여 단순히 Server-Client 방식으로 Nova, Neutron, Cinder와 같은 서비스가 설치된 서버의 상태를 측정할 뿐 서비스 프로세스의 상태는 측정할 수 없고 내부 인스턴스를 측정하려면 인스턴스마다 exporter를 설치해줘야 한다. 이는 사용자의 인스턴스에 직접 접근하는 방식으로 사용자의 개인 정보 보호가 취약할 수 있는 상황이 발생한다.
모나스카는 Monasca Agent를 설치한 서비스 서버에서 process.yaml를 통하여 각 서비스의 상태를 점검할 수 있고 libvirt.yaml를 통하여 Hypervisor 서버에서 가동하고 있는 VM(Virtual Machine)들의 정보를 수집하여 사용자 인스턴스의 접근할 필요 없이 손쉽게 정보를 얻어낼 수 있다.
셋째, 대시보드 면에서 프로메테우스는 자체적인 Web UI를 통하여 Service discovery에 저장되어있는 타겟들에게서 Exporter가 수집해온 메트릭을 기반으로 지표를 시각화 하거나 알림을 설정할 수 있다.
모나스카는 오픈스택 대시보드인 Horizon에서 모니터링 탭을 생성하여 알림을 설정할 수 있지만 시각적으로 지표를 보는 기능을 지원하지 않아서 지표를 분석해야하는 상황에서는 Grafana를 설치하여 시각적으로 나타내어야 한다.

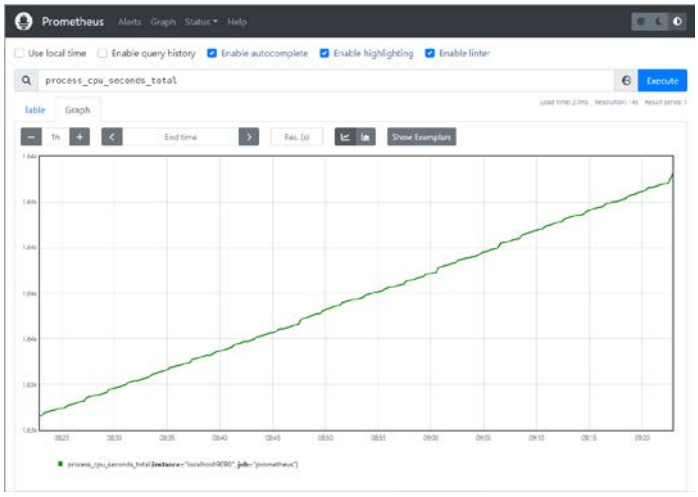

그림 3. 프로메테우스 Web UI

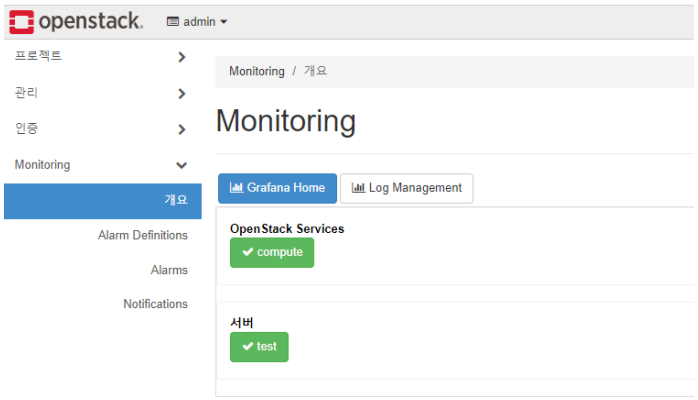

그림 4. Horizon에 Monitoring 탭이 추가된 모습

마지막으로 멀티테넌트에 관해서 프로메테우스는 오픈스택의 Keystone에 접근하지 못하여 모든 메트릭을 한 명의 관리자가 관리하는 방식이다. 따라서 각각의 테넌트 별로 구분하여 관리하는 것이 어렵다. 모나스카는 Keystone을 활용하여 오픈스택의 Tenant ID별로 메트릭을 저장 및 관측할 수 있어서 멀티테넌트 구조를 잘 지원한다. 이는 한 명의 관리자가 모든 것을 관리하는 것과는 달리 각 프로젝트 별로 관리자를 두어서 관리를 하게하는 오픈스택의 구조와 적합한 시스템이다.

4. 결론

본 논문은 오픈스택 환경에서의 모니터링 시스템 중 프로메테우스와 모나스카를 4가지 요소로 비교, 분석하였다. 이를 통해 프로메테우스에 비해 모나스카가 오픈스택 환경에서 필요한 기능을 다수 제공한다는 것을 알 수 있다. 그러나 이러한 기능을 사용하기 위해서는 어렵고 복잡한 설치 과정을 거쳐야 한다는 점이 큰 부담이다. 특히 전문 인력을 갖추기 어려운 중소 규모의 업체나 교육기관에서는 더더욱 중요한 부분이다. 따라서 오픈스택의 공식 프로젝트인 모나스카가 프로메테우스와 같이 높은 설치 편의성을 갖도록 개선해나가는 것이 필요하다고 판단된다.

참 고 문 헌

[1] OpenStack Document, www.openstack.org
[2] Grafana Document, https://grafana.com
[3] 안성열, 차병래, 차윤석, 전은진, 권귀영, 신병춘.(2021). 오픈소스 Prometheus 모니터링 시스템의 사전연구. 스마트미디어저널10(2),110-118.
[4] Prometheus Document, https://prometheus.io/
[5] Monasca Document, https://monasca.io