NFV 환경에서 Zabbix 모니터링 시스템을 활용한 VNF Auto-scaling

이지수, 염상길, 추현승
성균관대학교 소프트웨어대학
e-mail:{jisoo49, sanggil12, choo}@skku.edu

VNF Auto-scaling using Zabbix monitoring system in NFV environment

Jisoo Lee, Sanggil Yeom, Hyunseung Choo
College of Software, Sungkyunkwan University

요 약

최근 네트워크 서비스 관리의 복잡성을 줄이기 위해 새로운 네트워크 인프라가 등장하고 있다. NFV (Network Function Virtualization) 기술은 하드웨어 기반의 네트워크 장비에 가상화를 적용하여, 유연성 있는 네트워크 서비스를 제공한다. 네트워크 서비스는 Firewall, Parental Control (PC)과 같은 일련의 VNF (Virtual Network Function)로 구성된다. NFV 기술을 기존의 네트워크 환경과 통합시키는 경우 해결해야 할 난제가 존재한다. 기존 네트워크는 복잡성이 요구되며 많은 양의 트래픽을 다루어야 한다. 사용자가 요청한 네트워크 서비스의 높은 트래픽 로드로 인해 패킷 손실이 발생할 수 있다. 본 논문에서는 Zabbix 모니터링 시스템을 활용해 VNF 로드 기반의 Auto-scaling을 제안한다. 이를 통해 네트워크 서비스의 자원 효율성을 향상시키고 패킷 손실 비율을 줄일 수 있다.

1. 서론

오늘날 네트워크 트래픽 수요가 증가하면서 네트워크 공급자들은 NFV 기술을 통해 네트워크 인프라를 향상시키고자 한다. NFV는 가상화를 통해 하드웨어 장비로부터 네트워크 자원 및 서비스를 분리한다. [1] VNF는 Network Function (NF)의 네트워크 자원이나 서비스를 소프트웨어 기반으로 제공하는 기능이다. 이를 통해 다양한 네트워크 서비스를 비용 효율적으로 운영할 수 있다. [2] 그러나 기존의 네트워크 환경은 새로운 패러다임과 완전히 통합하는데 한계점을 가진다. 이전보다 복잡한 구조가 요구되며, 많은 양의 트래픽을 처리해야한다. 만약 네트워크 서비스를 구성하는 단일 VNF 인스턴스에 문제가 발생하면 전체 네트워크 서비스에 영향을 준다. 그 결과 높은 패킷 손실 비율과 VNF 인스턴스 로드 간의 불균형을 초래한다. 따라서 클라이언트의 요청이 들어오면 VNF 인스턴스를 적절하게 할당하는 것이 중요하다.

로드 밸런싱은 클라우드 컴퓨팅 환경에서 주로 요구되어지는 개념이다. 클라우드 환경에서는 클라이언트의 요청을 실행하기 위해 자원을 균형적으로 분배해야한다. 요청이 들어오면 가장 이용률이 낮은 VNF나 가상머신으로 할당함으로써 로드의 균형을 이룰 수 있다. 로드 밸런싱을 통해 자원을 효율적으로 활용할 수 있다.

NFV 환경에서도 요청에 따라 VNF 인스턴스를 로드 밸런싱 할 필요가 있다. Auto-scaling 기술은 VNF 로드를 자동으로 조절할 수 있다. 스케일링이란 요구되는 사용량에 따라 자원을 늘리거나 줄이는 동작을 말하며 수평적 스케일링과 수직적 스케일링으로 분류된다. 수평적 스케일링은 가상 머신의 수를 조절하는 것으로 scale-out과 scale-in이 있다. 수직적 스케일링은 가상 머신의 사양을 조절하는 것으로 scale-up과 scale-down이 있다.

Auto-scaling에 있어 기반이 되는 메트릭 정보(CPU, 메모리, 트래픽 사용량 등)는 모니터링 시스템을 통해 얻을 수 있다. 클라우드 모니터링 시스템인 Zabbix 오픈 소스를 활용하여 가상 자원을 실시간으로 모니터링 한다. 만약 오버로드나 결함이 발생할 경우, 이를 감지하여 알림을 발생시킨다. 본 논문에서는 NFV 환경에서 모니터링 시스템을 활용한 VNF Auto-scaling 기법을 제안한다.

관련 연구로 Zabbix를 활용하여 NFV 인프라의 장애를 감지하고 알림을 보내는 기법이 있다. [3], [4] 하지만 가상화 자원인 도커 컨테이너 (Docker container)를 위주로 모니터링한 결함 관리 구조를 제안한다. 또한 오픈 소스 모니터링 툴이 아닌 Tacker라는 기본 모니터링 방법을 사용한 VNF 상태 모니터링을 제안한다. [5]

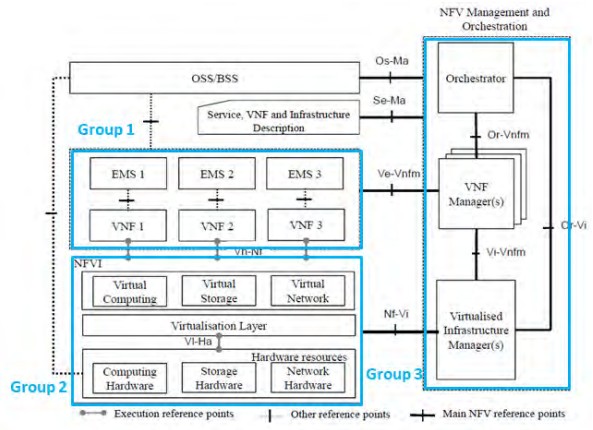
본 논문은 다음의 구성을 따른다. 먼저, 2장에서는 제안 기법에 바탕이 되는 관련 기존 연구들을 설명한다. 3장에서는 다양한 오픈 소스를 활용한 프레임워크를 제시하고 Auto-scaling 시스템 동작 과정을 설명한다. 마지막으로 4장에서는 결론과 향후 진행할 연구에 대해 설명한다.

2. 관련 연구

2.1 NFV Management and Orchestration (MANO)

그림 1은 ETSI에서 정의한 NFV 구조로 크게 3개의 기능 그룹으로 구성된다. [1] 첫 번째 그룹은 VNF와 VNF를 자체적으로 관리하는 Element Management Service (EMS)를 포함한다. 두 번째 그룹은 NFV Infrastructure (NFVI)이며 컴퓨팅, 스토리지, 네트워크 기능을 지원하는 물리적 자원과 가상화 계층을 통해 생성된 가상화 자원으로 구성된다. 세 번째 그룹인 NFV MANO는 물리적 자원, 가상화 자원, VNF를 관리한다.

NFV MANO의 주요 기능 블록은 Virtualized Infrastructure Manager (VIM), VNF Manager (VNFM), NFV Orchestrator (NFVO) 이다. VIM은 NFV의 물리적 자원과 가상화 자원을 관리하고 성능 측정을 위해 이를 수집한다. VNFM은 단일 또는 다수의 VNF 인스턴스에 배치되어 생명주기를 관리한다. NFVO는 다수의 VIM에 걸쳐 가상 자원을 통합적으로 조정하는 역할을 수행한다. 또한 네트워크 서비스의 인스턴스화, 스케일링, 초기화를 포함하는 생명주기를 관리한다. 이때 NFV MANO 환경에서 최종 사용자에게 신뢰성과 Quality of Service (QoS)를 보장하기위해 VNF 스케일링 기술을 적용할 수 있다.



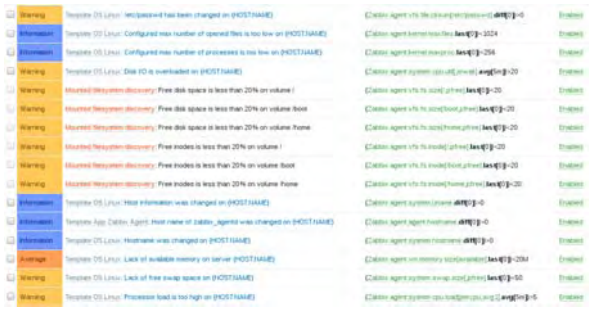
(그림 1) NFV 구조

2.2 Zabbix 모니터링 시스템

Zabbix는 엔터프라이즈 급 오픈소스 분산 모니터링 시스템이다. 네트워크의 수많은 매개변수와 서버의 상태에 대한 모니터링을 제공한다. [6] Infrastructure-level, Cloud environment-level, VM-level, Service-level와 같은 다양한 계층에서의 모니터링을 지원한다. Infrastructure-level은 인프라 역할을 하는 자원의 상태를 측정하는 계층으로 물리적 머신의 CPU, 메모리, 디스크 사용량 등을 측정한다. Cloud environment-level은 클라우드 환경의 상태를 측정하는 계층으로 클라우드 환경에 배치된 가상 머신의 수 등을 측정한다. VM-level은 가상 머신의 상태를 측정하는 계층으로 가상 머신의 컴퓨팅 리소스, CPU, 메모리, 스토리지 사용량을 측정한다. Service-level은 네트워크 서비스에 대한 성능 및 상태 정보를 측정한다.

Zabbix는 그림 2와 같이 이벤트 기반의 트리거를 구성할 수 있다. 트리거는 사용자로부터 정의된 임계값이 기준이 되며 임계값을 초과할 경우 사용자에게 알람을 전송한다. 알람은 SMS 또는 이메일 등의 실시간 통지 메커니즘으로 제공한다. 따라서 네트워크 문제가 발생할 경우 신속하게 판단하고 대처할 수 있다. 측정된 데이터는 웹 모니터링을 통해 어디서든 상태를 확인할 수 있다. 단일 패키지로 다양한 기능을 지원하는 Zabbix는 클라우드 환경에서 IT 자원을 모니터링 하는데 중요한 역할을 할 것이다.

Zabbix는 크게 에이전트와 서버로 구성된다. 에이전트는 모니터링 대상에 배포되어 물리적 머신 및 가상 머신의 자원을 모니터링 한다. 서버는 에이전트와의 상호작용을 통해 정보를 수집하고 트리거 및 알람 기능을 수행한다. 수집된 자원 상태는 Zabbix 웹 서버를 통해 통합적으로 모니터링 및 관리할 수 있다. NFVO MANO에서는 VNF를 모니터링하여 결함 발생 시 즉각적인 조치를 취해야하므로 Zabbix 모니터링 시스템을 필요로 한다.



(그림 2) Zabbix 이벤트 트리거

3. VNF Auto-scaling

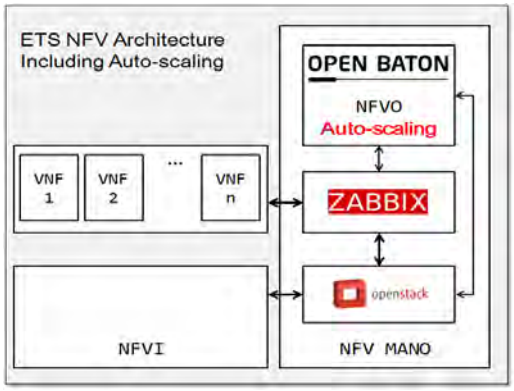
3.1 시스템 구조

그림 3은 다양한 오픈소스를 활용하여 Auto-scaling 기능을 수행하는 NFV 구조이다. ETSI NFV MANO 구조를 구현하기 위해 Open Baton 오픈 소스를 활용한다. Open Baton은 NFVO와 VNFM의 역할을 수행하며 다른 구성요소와 연결하기 위한 드라이버를 제공한다. [7] 모니터링 드라이버인 Zabbix 플러그인은 Zabbix와 Open Baton 간의 통신을 수행하며 서버 접속 없이 정보를 얻을 수 있다. Open Baton에서는 모니터링 된 VNF의 결함 및 성능을 확인한다. 또한, VIM 드라이버를 통해 VIM의 역할을 수행하는 OpenStack과 통신한다.

Open Baton은 NFV 환경에서의 Auto-scaling 시스템을 제공한다. Zabbix 플러그인을 사용해 원하는 정보의 모니터링이 가능하다. 그림 4의 Auto-scaling 정책은 VNF의 정보를 담고 있는 VNF Descriptor (VNFD)에 정의된다. Auto-scaling 정책으로 스케일링 조건을 정의하여 자동으로 VNF 인스턴스의 스케일링 동작을 할 수 있다. 정책은 이름, 조건, 액션, 주기 매개변수에 대한 내용으로 구성된다. 조건 매개변수는 알람 및 조건을 인식하는 모드, 측정

결과와 비교할 임계값 그리고 비교 연산자를 포함한다. 해당 모니터링 메트릭이 조건을 충족하면 알람 신호가 생성되며 액션을 수행한다. Auto-scaling 시스템에서 실행 가능한 스케일링 액션은 다음과 같다.

SCALE_OUT: 인스턴스의 특정수를 확장
SCALE_IN: 인스턴스의 특정수를 축소
SCALE_OUT_TO: 인스턴스의 특정수로 확장
SCALE_IN_TO: 인스턴스의 특정수로 축소



(그림 3) Auto-scaling을 수행하는 NFV 구조

```
"auto_scale_policy":[
  { "name":"scale-out",
    "threshold":100,
    "comparisonOperator":">=",
    "period":30,
    "cooldown":60,
    "mode":"REACTIVE",
    "type":"WEIGHTED",
    "alarms":[
       {
        "metric":"system.cpu.load[percpu,avg1]",
        "statistic":"avg",
        "comparisonOperator":">",
        "threshold":0.70, "weight":1
       }
    ]
  }
]
```

(그림 4) Auto-scaling 정책

3.2 동작 원리

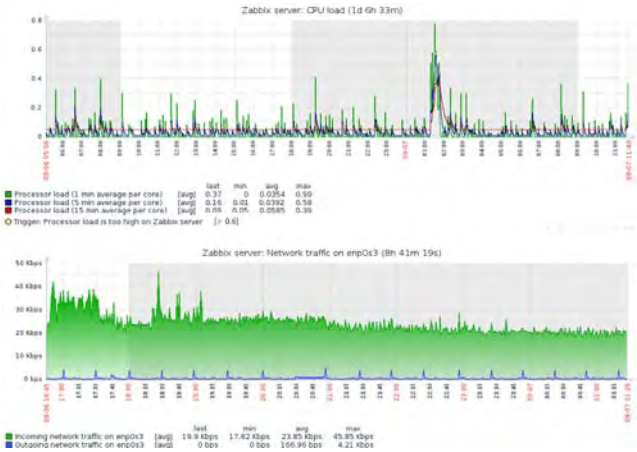
Auto-scaling의 동작 원리는 3단계로 구분된다. 첫 번째는 스케일링 동작 여부를 판단하는 모니터링 단계이다. 두 번째 단계는 상태에 따라 스케일링 동작을 판단하는 결정 단계이다. 세 번째 단계는 결정 단계를 기반으로 스케일링 동작을 행하는 실행 단계이다.

첫 번째 모니터링 단계는 VNF 상태를 모니터링하고, 조건에 따라 알람을 발생시킨다. 모니터링 시스템으로는 Zabbix를 활용한다. VNF 인스턴스의 정보를 나타내는 VNF Descriptor (VNFD)에는 이러한 알람 조건을 포함한다. 각 VNF의 알람 조건을 표현할 때에 고려해야할 사항이 있다. 먼저, VNF의 특성을 고려하여 메트릭을 선정해야 한다. 예를 들어, CPU와 연관성이 큰 VNF라면 CPU 로드를 모니터링 메트릭으로 선정한다. 또한, 해당 메트릭의 임계값을 몇으로 할지 고려해야한다. 예를 들어, VNF의 CPU 로드를 모니터링 할 때 평균값과 패킷 손실이 발생하는 값을 고려하여 설정한다.

두 번째 결정 단계는 알람을 기반으로 스케일링 동작을 결정한다. 모니터링 단계에서 특정 트리거가 발생하면 NFVO는 어떤 조치를 취해야 하는지 판단한다. 문제 유형에 따른 스케일링 동작을 결정할 뿐만 아니라 시스템에서 실현 가능한 동작인지를 판단한다. 예를 들어, scale-out은 추가적으로 인스턴스를 구성할 자원이 충분한지 고려해야 한다. 구체적인 결정 단계를 위해서는 Auto-scaling 정책에 추가적인 조건과 로직을 구현해야한다.

세 번째 실행 단계는 결정 단계에서 요구하는 스케일링 동작을 수행한다. NFVO는 동작에 대한 실행을 요청하며 각 요청은 스케일링 동작 리스트를 포함한다. VIM은 자원을 할당하거나 배포하는 스케일링 동작을 실행한다. 만약 동일한 요청이 들어오면 실행하지 않는다.

동작 과정은 높은 로드의 VNF 인스턴스에 대한 Auto-scaling 시나리오를 기반으로 한다. 네트워크 서비스를 제공하는 VNF의 로드가 매우 높아 패킷 손실이 발생한다. 그림 5에서처럼 Zabbix는 CPU 로드와 트래픽 로드 상태에 대한 모니터링 제공한다. CPU 로드가 임계값인 0.6 이상이 되면 트리거가 발생한다. Auto-scaling 시스템은 Zabbix로부터 알람을 수신하여 NFVO에게 scale-out 동작을 요청한다. NFVO는 VNFM에게 새로운 VNF 인스턴스 할당을 요청한다. VIM과 VNFM는 VNF 인스턴스를 생성하고 NFVO에게 응답한다. 네트워크 서비스는 로드가 낮은 VNF 인스턴스가 할당되어 로드 밸런싱을 이룬다.



(그림 5) Zabbix를 통한 CPU 로드(상), 네트워크 트래픽 로드(하) 모니터링

4. 결론 및 향후 연구

본 논문에서는 Zabbix 모니터링 시스템을 활용한 VNF 인스턴스의 Auto-scaling 기법을 제안한다. 기존의 네트워크 환경에서 NFV 기술을 통합할 때 발생하는 난제를 해결한다. 이때 모든 시스템 관리에 기본인 모니터링을 활용

한다. Zabbix 모니터링 시스템은 VNF 로드를 모니터링 하여 미리 정의된 조건에 따라 알림을 보낸다. 알람에 따라 새로운 VNF를 할당하거나 제거하는 스케일링 동작을 수행한다. 이를 통해 네트워크 서비스를 구성하는 VNF 간의 로드 밸런싱을 이루며 패킷 손실 비율을 줄인다. 프레임워크는 NFV MANO 기능의 Open Baton 기반으로 구성되며 추가적인 구성 요소는 기타 오픈 소스들을 활용한다. Zabbix의 모니터링 동작 검증은 완료하였으며, 추후 연구를 통하여 제시한 Auto-scaling 시스템 구조에 적용시킬 예정이다. 또한 각 VNF의 특성을 고려한 로드 및 조건 선정에 있어 심층적으로 연구하고자 한다.

ACKNOWLEDGEMENT

본 논문은 기초연구사업 (NRF-2010-0020210)과 과학기술정보통신부 및 정보통신기술진흥센터의 Grand ICT연구센터지원사업 (IITP-2017-2015-0-00742), 방송통신인프라 원천기술개발사업 (B0101-17-1366, 자율 제어 네트워킹 및 자율 관리 핵심 기술 관리)의 연구결과로 수행되었음

참고문헌

[1] ETSI, “Network Functions Virtualisation (NFV); Architectural Framework,” ETSI GS NPV 002 V1.2.1, December 2014.

[2] 오유미, and 이성원. "오픈 네트워크 플랫폼 기반 오픈소스의 NFV 를 통한 VNF 개발에 관한 연구." 한국통신학회 학술대회논문집, pp. 1394-1395, 2015.

[3] 이한범, and 김화성, "Zabbix를 활용한 NFV 인프라 장애 감지 및 알림 방안 연구." 한국통신학회 학술대회논문집, pp. 1171-1172, 2016.

[4] 이한범, and 김화성, "Zabbix 기반의 VNF 모니터링 기법." 한국통신학회 학술대회논문집, pp. 1163-1164, 2016.

[5] 김민욱, and 김영한. "오픈스택 NFV 환경에서 VNF 서비스 모니터링을 위한 다양한 오픈소스 모니터링 툴과의 연동 구조 설계." 한국통신학회 학술대회논문집, pp. 125-126, 2017.

[6] https://en.wikipedia.org/wiki/Zabbix

[7] Open Baton. [Online]. Available: http://www.openbaton.github.io/