K-ONE 기술 문서 #40

A Design of M-CORD Anomaly Detection

Document No. K-ONE #00
Version 1.0
Date 2019-08-31
Author(s) 홍지범

K-ONE

ONK
OpenNetworkingKorea

■ 문서의 연혁

버전	날짜	작성자	내용
Draft - 0.1	2019. 08. 16	홍지범	초안 작성
Draft - 0.2	2019. 08. 23	홍지범	설계 내용 추가
Draft - 1.0	2019. 08. 31	홍지범	오탈자 수정

이 논문은 2019 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2015-0-00575, 글로벌 SDN/NFV 공개소프트웨어 핵심 모듈/기능 개발)

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2015-0-00575, Global SDN/NFV OpenSource Software Core Module/Function Development)

기술문서 요약

본 기술문서는 M-CORD 환경에 구성된 모니터링 시스템을 이용한 비정상 상태 탐지 (Anomaly Detection) 서비스에 대한 디자인을 서술한다. CORD는 모바일 네트워크를 엣지 클라우드에서 서비스하기 위한 플랫폼으로, ONF에서 개발 중인 프로젝트이다. 이 플랫폼 내에는 EPC 네트워크를 구성하는 다양한 VNF들이 구동이 되고 있다. M-CORD 모니터링 시스템은 각 VNF가 사용하는 컴퓨팅 및 메모리 자원과 VNF들이 송/수신하는 데이터 트래픽을 모니터링 하는 시스템이다. 모니터링 시스템을 통해 M-CORD 플랫폼 상에서 구동 중인 VNF들이 얼마나 많은 컴퓨팅 자원을 소모하고, 얼마나 많은 데이터 트래픽을 송/수신하는지를 모니터링할 수 있다. 그리고 비정상 상태 탐지 서비스는 EPC 네트워크를 구성하는 각 VNF들이 정상적으로 동작하는지를 기계학습으로 생성된 모델을 통해 판별한다. 본 기술문서에서는 이러한 M-CORD의 모니터링 시스템을 이용한 비정상 상태 탐지 서비스를 구현하기 위한 디자인을 제안한다.

Contents

K-ONE #00. A Design of M-CORD Anomaly Detection

그림 목차

K-ONE #40. A Design of M-CORD Anomaly Detection

1. Introduction

기존 네트워크 환경에서 제공되고 있던 서비스들은 최근 하드웨어 및 소프트웨어 자원 활용을 최적화하고, 서비스를 보다 유연하고 효율적으로 관리하기 위해 SDN (Software-Defined Networking), NFV (Network Function Virtualization) 및 클라우드 컴퓨팅 (Cloud Computing) 기술과 결합하여 사용자에게 제공되고 있다. CORD (Central Office Re-architected as a Data Center) 는 미국 ONF (Open Networking Foundation)에서 현재 연구/개발이 진행 중인 데이터센터 플랫폼이다. CORD는 기존 통신사업자 (SK Telecom, KT, AT&T 등)들이 운용하는 전화국 (Central Office; CO)을 데이터센터로 변경하여 운용하는 개념이다. 기존 통신사업자의 전화국에서 운용되는 네트워크 장비들을 가상화하여 CORD 플랫폼 내 서버에서 동작시키고, 이 외에도 다양한 종류의 가상 네트워크 기능들 (Virtual Network Function; VNF) 혹은 어플리케이션들을 구동한다. 이에 따라 엣지 클라우드 컴퓨팅 뿐만 아니라 다양한 서비스들을 사용자에게 제공할 수 있다.

CORD 플랫폼은 사용 목적에 따라 Residential CORD (R-CORD), Enterprise CORD (E-CORD), 그리고 Mobile CORD (M-CORD)로 구분된다. R-CORD는 댁내 사용자들의 네트워크를 CORD 플랫폼을 통해 연결하는 것이다. E-CORD는 회사 내부 네트워크를 CORD 플랫폼으로 구성하는 것이다. 마지막으로 M-CORD는 통신사업자가 기존에 운용하는 모바일 네트워크를 CORD 플랫폼을 통해 서비스를 제공하는 것을 목표로 한다.

M-CORD는 LTE 및 5G EPC에서 사용하는 네트워크 기능 (Serving Gateway, Packet Gateway, Mobility Management Entity 등)을 가상화하고, CORD 플랫폼 내부 서버에서 동작시킨다. 이후, 외부에 설치된 기지국 (evolved NodeB; eNB)들을 M-CORD 플랫폼에서 동작하는 VNF로 연결하여 M-CORD를 통해 모바일 네트워크를 구축한다. 또한 M-CORD는 하나의 데이터센터 플랫폼이기 때문에 새로운 어플리케이션 및 다른 VNF들을 동작시키는 것이 가능하다. 따라서 모바일 엣지 컴퓨팅 (Mobile Edge Computing; MEC) 혹은 다양한 어플리케이션을 제공할 수 있을 것으로 기대된다.

하지만 M-CORD는 연구 및 개발이 진행 중인 프로젝트이기 때문에, 현재 데이터센터 플랫폼 운용에 필요한 모든 기능들을 제공하지 않는다. 특히 M-CORD에서 운용되는 다양한 노드 및 VNF들의 컴퓨팅 리소스 사용량 및 네트워크 트래픽 사용량과 같은 기본적인 정보를 제공하는 모니터링 시스템을 제공하지 않는다. 이에 따라 관리자는 M-CORD의 성능 조정 및 최적화와 같은 관리를 위한 의사결정에 어려움을 겪을 수 있다.

본 기술문서에서는 이러한 문제를 해결하기 위해 M-CORD 플랫폼을 모니터링하는 시스템을 구축한다. 또한 구축된 모니터링 시스템을 이용해 M-CORD 플랫폼 내에서 동작하는 물리 서버 및 VNF들의 비정상적인 동작 상태를 탐지하는 서비스에

대한 디자인을 제시한다. 이를 통해 M-CORD 플랫폼에서 동작하는 서버 및 VNF들의 CPU 및 메모리 부족과 같은 리소스 사용에 대한 비정상 상태를 탐지하여 관리자가 자원 최적화 및 성능 조정에 대한 의사결정을 하는데 도움을 줄 수 있다.

본 기술문서의 구성은 다음과 같다. 먼저 2절에서는 CORD 및 M-CORD의 특징과 구조에 대해 서술한다. 다음으로 3절에서는 M-CORD 모니터링 시스템의 디자인 및 구현에 대해 서술한다. 마지막으로 4절에서는 M-CORD 모니터링 시스템을 이용한 비정상 상태 탐지 서비스의 디자인을 제시하고 5절에서 결론을 맺는다.

2. CORD

본 절에서는 CORD가 등장한 배경과 필요성, 도입 효과에 대해 설명하고, CORD 아키텍쳐 구조에 대해 설명한다.

2.1. CORD 개요

기존 네트워크 환경에서 통신 사업자는 전화 통신 및 패킷 통신을 위해 전화국을 설치하고, 전화국에서 네트워크 장비를 통해 통신 교환을 수행하였다. 하지만 기존에 사용되던 네트워크 장비들은 전용 장비 (dedicated hardware)를 사용하기 때문에 폐쇄적인 (closed) 특징을 지닌다. 이와 같은 폐쇄적인 전용 장비들은 운용에 있어 다음과 같은 몇 가지 문제를 야기한다. 먼저 범용 장비 (commodity hardware)가 아닌 전용 장비를 사용하기 때문에 장비를 구매하는데 많은 비용이 소모되며, 더 많은 장비가 필요한 경우 소프트웨어를 통해 기능을 확장하는 것이 아니라 전용 장비를 추가로 구입해야 한다.

또한 기존 기능의 수정이나 새로운 기능이 요구되는 경우, 네트워크 운영자가 아닌 장비 제조사에서 수정 및 개발을 해야 하기 때문에 시간과 비용이 많이 소요된다. 이러한 이유로 통신 사업자는 기존 네트워크 장비들의 구매 및 운용비용 (CAPEX/OPEX)이 많이 든다는 단점이 있다. 예를 들어 미국 AT&T의 경우 미국 전역에 전화국이 약 4,700개가 설치되어 있으며, 각 전화국에는 최소 300개 이상의 폐쇄적인 전용 장비들이 운용되고 있기 때문에 많은 비용을 지출하고 있다.

이러한 문제를 해결하기 위해 등장한 것이 CORD (Central Office Re-architected as a Data Center) 프로젝트이다. CORD의 주요 목적은 기존 전화국을 데이터센터로 변경하여 운용하는 것을 목표로 한다. 기존 전화국이 폐쇄적인 전용 장비를 사용했다면, CORD는 개방적인 범용 장비 (commodity hardware)를 사용한다. 이러한 범용 장비를 활용하여 전화국의 기존 역할을 그대로 수행할 수 있도록 한다. 이를 위해 전화국에서 운용되는 다양한 네트워크 기능들을 가상화하고, 가상화된 VNF들을 범용 장비에서 동작시킨다. 이처럼 개방적인 범용 장비들을 사용할 경우, 장비 제조사의 종속성 관계에서 벗어날 수 있다. 또한 새로운 기능이 요구될 때 운영자가 직접 원하는 기능을 VNF 또는 범용 장비에 구현할 수 있게 되어 개방적인 범용 장비를 사용함으로써 비용을 절약할 수 있다는 장점이 있다.

따라서 CORD 플랫폼을 사용하면 통신 사업자는 기존에 운용하는 전화국을 새로운 데이터센터로 사용을 할 수 있다. 또한 통신 사업자 및 네트워크 사업자들은 CORD를 활용하여, 5G 모바일 네트워크 및 기타 유선 네트워크 서비스들을 제공할 수 있다. 마지막으로 장비 제조사 혹은 서비스 제공업체들은 CORD 플랫폼을 모바일 엣지 컴퓨팅 기술을 활용할 수 있는 엣지 클라우드 (edge cloud)로 사용하는 것이 가능하다.

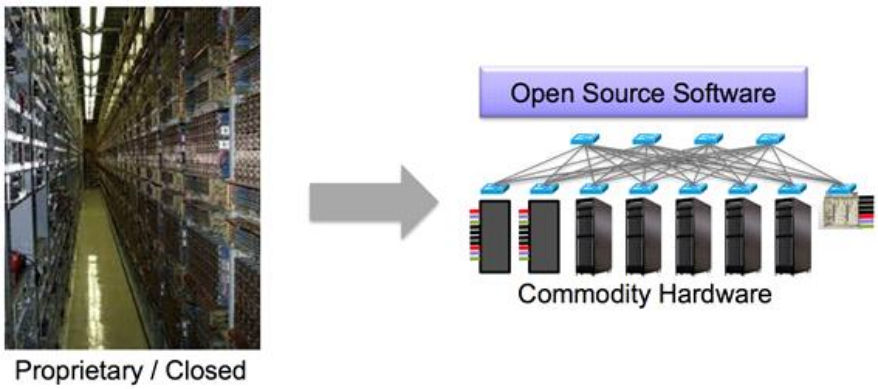

그림 1. Vision of CORD project

그림 1은 CORD 프로젝트의 비전을 나타낸다. 먼저 좌측에 있는 그림은 기존 전화국의 모습으로, 전화국 내부에 폐쇄적인 전용 장비들이 무수히 많이 설치되어 있는 것을 볼 수 있다. 이와 반대로 오른쪽 그림은 CORD의 구조를 나타내는 그림으로, CORD는 시중에서 구매할 수 있는 범용 장비를 활용하여 데이터센터를 구축하였으며, 플랫폼 내부에서 동작하는 VNF는 오픈소스 소프트웨어 (Open-Source Software)로 구현한다. 따라서 운영자는 CORD를 통해 보다 적은 비용으로 네트워크를 구축할 수 있으며, 플랫폼 내부에서 동작하는 VNF 및 소프트웨어를 자유롭게 수정 및 추가를 할 수 있는 장점이 있다.

CORD 플랫폼은 SDN (Software-Defined Networking), NFV (Network Function Virtualization), 그리고 클라우드 컴퓨팅 (Cloud Computing) 기술을 사용된다. SDN은 네트워크의 기능을 제어평면과 데이터평면으로 분리하여 제어 평면에 위치한 소프트웨어 컨트롤러를 통해 네트워크를 제어 및 관리하는 기술을 의미하며, NFV는 폐쇄적인 전용 장비를 통해 동작하는 네트워크 기능들을 가상화하여 범용 장비에서 VNF 형태로 동작시키는 기술을 의미한다. 마지막으로 클라우드 컴퓨팅 기술은 사용자 혹은 네트워크 관리자가 제공하는 서비스에 대해 그 수요에 따라 가상 머신 (VM) 등의 자원 할당 및 서비스 인스턴스의 수를 탄력적으로 조정하여 운용하는 기술을 의미한다.

이러한 기술들을 통해 CORD에서는 상용 SDN 네트워크 장비를 사용하여 데이터센터 네트워크를 제어 및 관리할 수 있으며, 다양한 전용 장비들을 데이터센터에 위치한 범용 서버에서 VNF의 형태로 운용할 수 있다.

2.2. CORD Architecture

CORD는 그림 2와 같은 구조를 갖는다. 먼저 CORD 플랫폼을 전체적으로 제어 및 관리하는 컨트롤러인 XOS가 존재한다. CORD는 서비스 제어 평면 (Service control plane) 역할을 하는 XOS를 이용하여 CORD 플랫폼이 포함하는 다양한 구성요소를 제어 및 관리한다. XOS가 관리하는 구성 요소들은 ONOS 컨트롤러가 있고, OpenStack 또는 Kubernetes, 그리고 하드웨어이다. 이러한 구성요소들은 서비스 데이터 평면 (Service data plane)의 역할을 한다.

먼저 ONOS 컨트롤러는 CORD 플랫폼을 구성하는 하드웨어 중 화이트 박스 스위치 (white box switch)를 제어하는 오픈소스 소프트웨어이다. ONOS 컨트롤러는 다양한 어플리케이션 (Ctrl App)들을 통해 화이트 박스 스위치의 동작을 제어한다. 다음으로 OpenStack 또는 Kubernetes는 서버 (Compute node) 내의 가상 머신/컨테이너를 제어 및 관리하는 역할을 하는 오픈소스 소프트웨어이다. 이를 통해 가상 인프라를 구축함으로써 사용자가 원하는 VNF들을 서버에 배포할 수 있는 특징이 있다. CORD 플랫폼에 사용되는 하드웨어는 OCP (Open Compute Project)에 속한 하드웨어를 일반적으로 사용한다고 가정한다. 이 하드웨어들은 앞서 언급한 ONOS 컨트롤러로부터 네트워크를 제어 받거나 CORD 플랫폼 내 서버 상에 배포된 다양한 VNF들을 구동하는 역할을 수행한다.

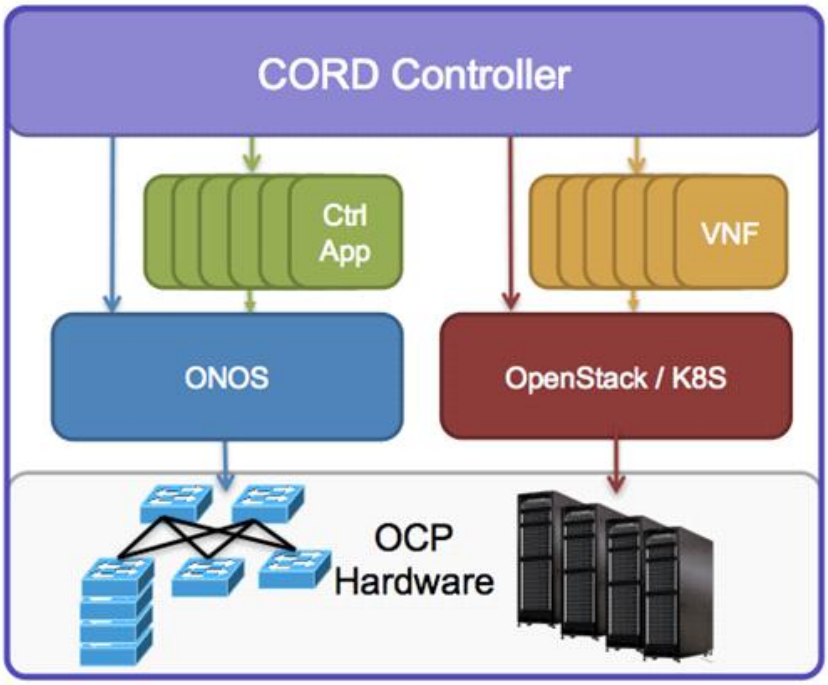

그림 2. Overall architecture of CORD

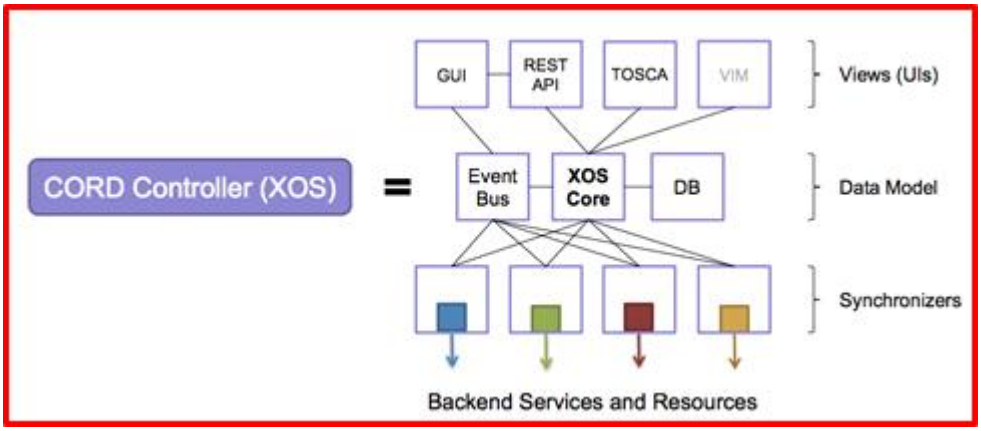

그림 3. Architecture of CORD controller (XOS)

위의 구성 요소들 중, CORD 전체를 제어/관리하는 부분인 XOS는 그림 3과 같이 뷰 (View), 데이터 모델 (Data model), 그리고 싱크로나이저 (Synchronizer)의 계층으로 구분되어 있다. 뷰 계층은 사용자 혹은 운영자와 CORD가 상호작용을 하는 UI를 의미하며, GUI, REST API, TOSCA 등이 포함되어 있다. 데이터 모델 계층은 각 서비스들의 모델을 의미하며, 사용자가 원하는 서비스를 제공받기 위해 CORD 플랫폼 내에서의 수행 절차 및 서비스 특징과 같은 다양한 정보들이 모델링되어 있다. 마지막으로 싱크로나이저 계층은 모델링된 정보를 서비스 데이터 평면에 전달하여 서비스가 수행되도록 하는 역할을 한다. 이를 위해 싱크로나이저는 서비스마다 하나씩 수행이 되며, 사용자 및 CORD 플랫폼내부의 시스템의 요청을 인지하도록 주기적으로 폴링 (polling)을 하고 있다.

2.3. CORD Configuration

CORD를 구성하기 위해서는 그림 4와 같이 크게 물리적인 파드 (Physical POD)를 구성하는 방법과 하나의 서버에서 CORD를 구성하는 CiaB (CORD-in-a-Box) 2가지 방법이 있다. 먼저 물리적 파드를 구성하는 방법은 실제로 CORD 플랫폼을 운용할 수 있는 수준의 성능을 가진 물리적 장치들을 갖추어 구성하는 것이다. 이를 위해 그림4의 좌측 그림과 같이 1개의 관리 스위치(Management switch), 4개의 패브릭 스위치 (Fabric switches), 그리고 3대의 X86 서버가 필요하다.

관리 스위치는 각 장비들을 제어 및 관리하기 위한 관리 네트워크 (Management network)를 의미하며, 패브릭 스위치들은 서버 (Compute node)들 사이의 사용자 데이터 트래픽 (User data traffic)을 구성하기 위한 스위치이다. CORD는 리프-스파인 (Leaf-Spine) 패브릭 구조로 서버를 연결한다. 그리고 3대의 X86 서버들 중 한 대는 CORD 전체를 제어 및 관리하기 위해 다양한 컨트롤 서비스들이 정의된 헤드 노드 (Head node)로써 동작한다. 나머지 2대는 사용자들이 사용하는 VNF들을 구동하기 위한 컴퓨트 노드 (Compute node)로써 동작한다.

헤드 노드에는 컴퓨트 노드 및 스위치를 제어 및 관리하기 위해 관리 네트워크를 통해 다른 모든 노드에 접근이 가능하다. 헤드 노드에서 구동 중인 컨트롤 서비스는 XOS, ONOS, XOS, MAAS, OpenStack/Kubernetes 등으로 이루어져 있다.

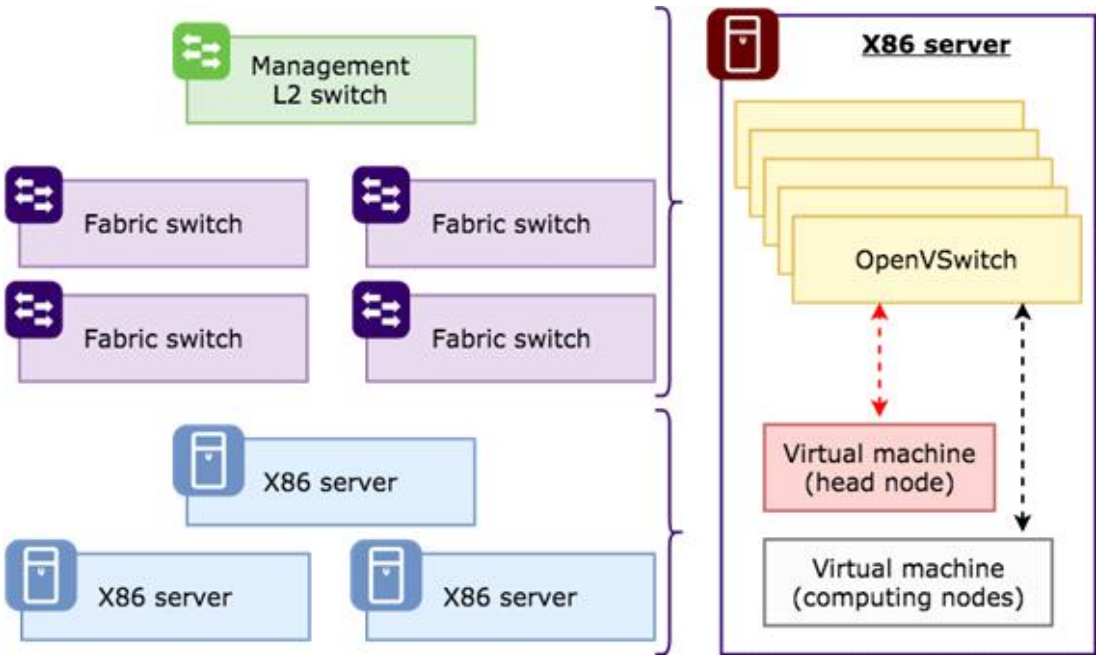

그림 4. CORD configuration (physical POD & CiaB)

이러한 서비스들은 CORD의 전반적인 관리와 더불어 컴퓨트 노드의 설정이나 전원 관리, CORD를 구성하는 스위치들의 라우팅 및 포워딩을 제어하고, VNF들이 동작 중인 가상 네트워크를 관리한다.

하지만 물리적 파드를 구성하여 CORD의 실험 및 개발을 하려면 물리적인 하드웨어를 구입하는 것부터 많은 비용이 들게 된다. 이를 해결하기 위해 CORD 프로젝트에서는 그림 4 우측 그림과 같이 개발자 및 연구원들을 위한 CiaB를 제공한다. CiaB는 CORD-in-a-Box의 약자로 물리적 파드에서 서비스하는 모든 물리적 장치를 한 대의 X86 서버에서 CORD 환경을 구성하는 것을 의미한다.

관리 스위치의 에뮬레이션에는 X86 서버 내에 브릿지를 만들고, 스위치를 에뮬레이션 하기 위해 2개의 OpenvSwitch를 이용한다. 마지막으로 헤드 노드와 컴퓨트 노드를 에뮬레이션에는 X86 서버에 가상 머신을 구동하는 방법을 이용한다.

3. M-CORD

본 절에서는 CORD의 여러 종류 중 하나인 M-CORD의 등장배경 및 개념과 전반적인 구조에 대해 설명한다.

3.1. M-CORD 개념

기존 모바일 네트워크 서비스를 위한 LTE 네트워크는 중앙 집중식 구조를 가졌다. 기존 모바일 네트워크는 크게 무선 네트워크인 EUTRAN (Evolved UTRAN)과 코어 네트워크인 EPC (Evolved Packet Core)로 나뉜다. EUTRAN은 실제 사용자 휴대폰과 기지국 사이의 네트워크를 의미하고, EPC는 기지국으로부터 온 사용자 데이터를 인터넷 혹은 다른 패킷 네트워크 (Packet Data Network; PDN)로 전달하거나 사용자의 인증을 하는 역할을 한다. 이를 위해 EPC 네트워크 내에서는 사용자의 인증 및 제어를 위한 MME (Mobility Management Entity), HSS (Home Subscriber Server) 등이 있으며, 데이터를 전달하기 위한 S-GW (Serving Gateway)와 P-GW (PDN Gateway)가 있다. 이 때 기존 EPC 네트워크는 중앙에 위치하며, 통신 사업자가 EPC의 각 네트워크 기능들을 관리하고 있다.

M-CORD는 기존 모바일 EPC를 구성하는 MME, HSS 등과 같은 다양한 네트워크 기능 (Network function)들을 CORD 플랫폼 내에서 구동하는 것이다. 그림 5는 M-CORD의 개념도를 나타낸다. M-CORD는 CORD 플랫폼 외부에 설치된 기지국에서 사용자 트래픽을 수신한다. 이 때, 기존 모바일 네트워크는 중앙에 위치한 EPC로 사용자 트래픽이 전달되지만, M-CORD의 경우 중앙에 위치한 EPC 네트워크가

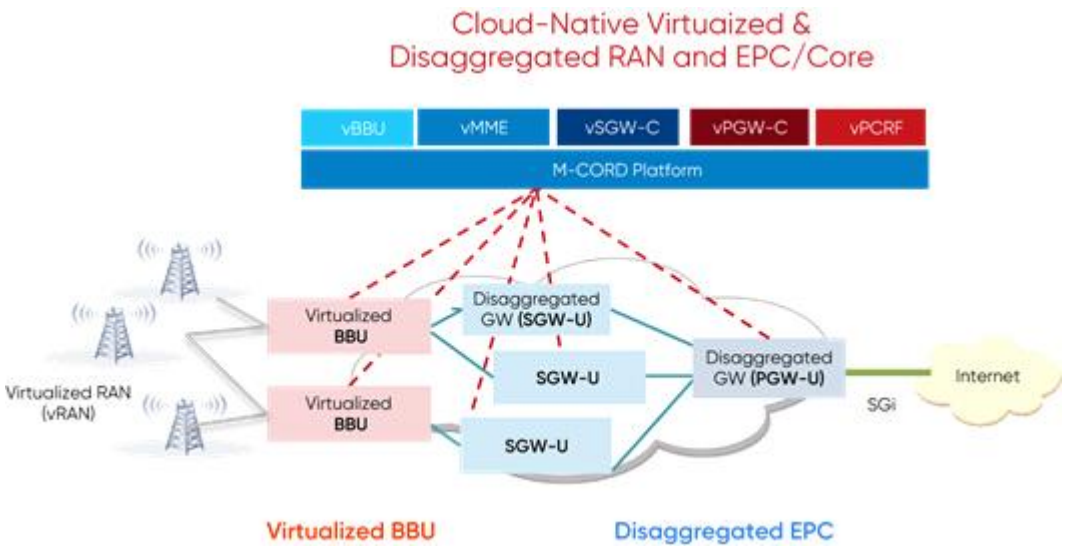

그림 5. Overview of M-CORD

아닌 엣지 클라우드 (edge cloud)에 위치한 분산된 EPC 네트워크로 사용자 트래픽을 전달한다. 이와 같이 전국에 퍼져있는 전화국을 데이터센터로 재구성하고, 중앙이 아닌 분산된 EPC 네트워크에서 NF를 동작시킨다면 중앙에 위치한 EPC 네트워크의 부하가 분산되는 효과가 있다. 또한 사용자 트래픽이 EPC 네트워크까지 도달하는 시간을 단축할 수 있는 효과가 있다.

3.2. M-CORD의 구조

CORD 플랫폼에서 모바일 EPC 네트워크의 NF를 구동하려면 먼저 EPC 네트워크에서 동작하는 NF을 소프트웨어화 (Softwarization) 및 가상화 작업이 필요하다. 그림 6은 기존의 EPC 네트워크의 구성요소를 나타낸다. 기존 EPC는 내부의 NF들을 전용 장비를 이용하여 구동하였다. 하지만 이는 앞서 언급한 바와 같이 많은 유지비용과 구매비용이 필요할 뿐만 아니라 데이터센터에서 활용이 가능하도록 범용 장비를 통해 구동하기가 어렵다. 따라서 M-CORD에서는 기존 EPC 네트워크의 NF들을 오픈소스 형태의 소프트웨어로 구현하고, 그림 6과 같이 이를 가상 머신 (Virtual Machine; VM)이나 컨테이너 (Container)의 형태로 가상화시킨 VNF를 이용한다.

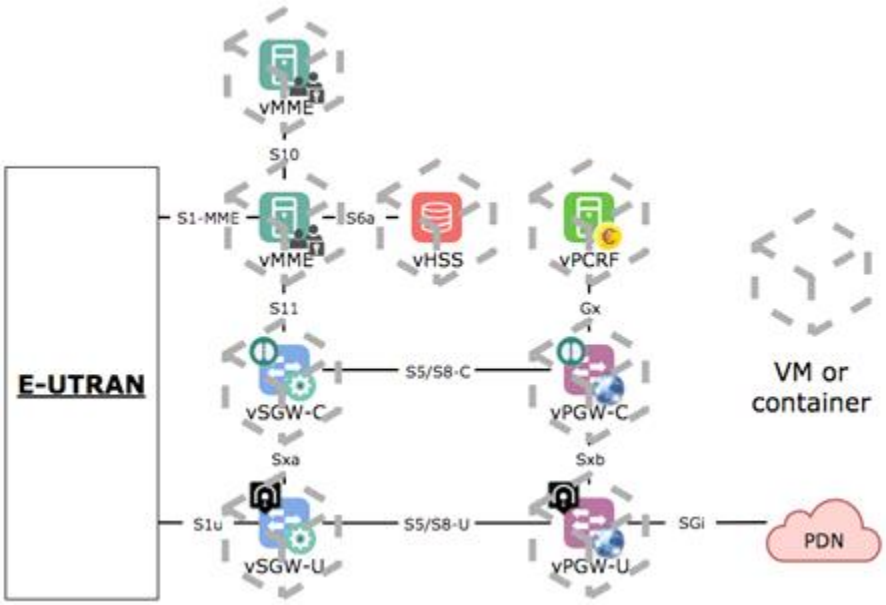

그림 6. M-CORD EPC network

M-CORD는 버전에 따라 다른 VNF를 제공한다. 4.1 버전의 경우 S-GW와 P-GW를 SPGW로 통합한 후, 제어 평면만 분리한 SPGW-C와 사용자 평면만 분리한 SPGW-U가 있다. 그리고 SPGW-C와 SPGW-U의 동작을 검증하기 위한 eNB 에뮬레이터가 존재한다. eNB 에뮬레이터는 NG4T사에서 개발된 소프트웨어를 사용하며, 내부에는 MME, HSS, eNB, 그리고 인터넷을 에뮬레이션할 수 있도록 설계되어있다. 각 VNF를 연결하는 가상 네트워크의 이름은 LTE 표준 인터페이스를 참조하여 명명되었다.

5.0 버전부터는 4.1 버전과 동일하게 SPGW-C와 SPGW-U가 있으며, 4.1 버전에서 eNB 에뮬레이터를 통해 에뮬레이션 하던 것을 실제 VNF로 대치하였다. 이전 버전에서 세션 관리 (session management) 등의 경우 MME와 HSS를 에뮬레이션하는 것이 아닌 실제 사용이 가능한 VNF로 구동시켰다. 또한 기존 eNB 에뮬레이터가 에뮬레이션 하던 EUTRAN 트래픽은 실제 eNB 하드웨어 및 휴대폰을 사용하도록 대치되었다.

4. M-CORD Monitoring System

본 절에서는 M-CORD 환경을 모니터링하기 위한 시스템의 디자인 및 구현에 대해 설명한다.

4.1. Design

3절에서 언급한 M-CORD 환경에서, M-CORD를 위한 모니터링 시스템을 디자인하고 구현하였다. 그림 7은 M-CORD 모니터링 시스템에 대한 디자인 개념도이다. 현재 CORD 5.0 이상 버전의 환경을 테스트하기 위한 하드웨어 eNB의 제약으로 인해 M-CORD 4.1 버전을 기반으로 디자인 및 구현되었다. M-CORD 내에는 앞절에서 설명한 CORD 구조와 같이 헤드 노드와 컴퓨트 노드들로 구성되어 있다. 컴퓨트 노드에는 M-CORD에서 사용하는 다양한 VNF들이 Docker 기반의 컨테이너 형태로 구동된다. VNF들 사이의 네트워크 인터페이스는 가상 인터페이스를 사용하여 연결된다.

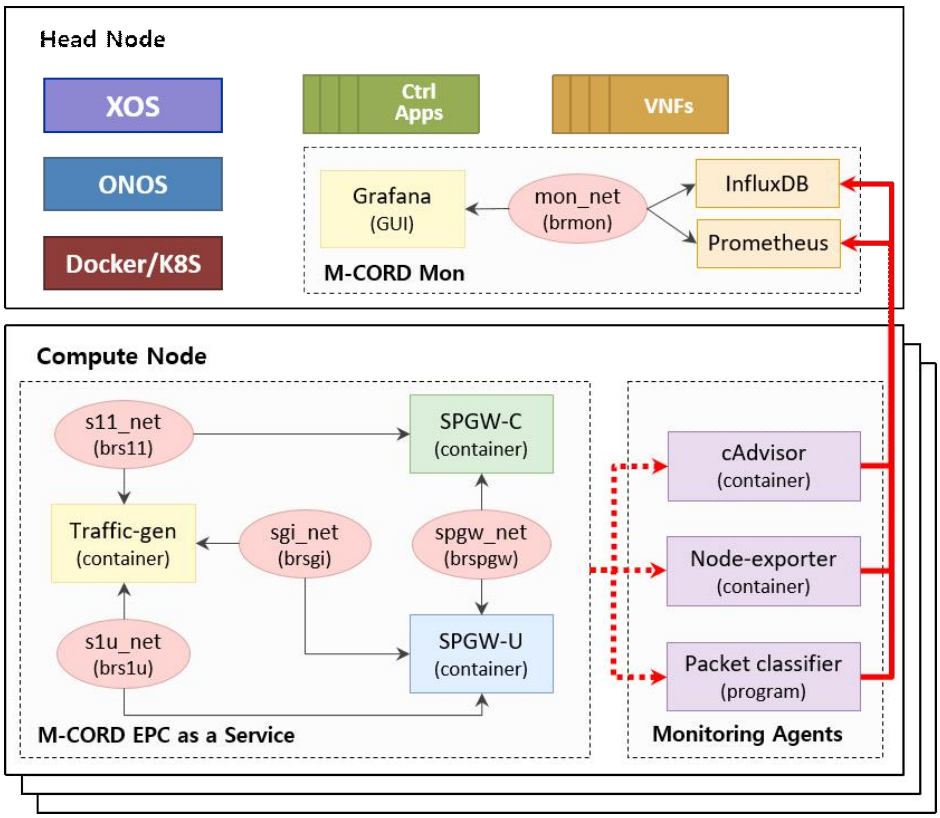

그림 7. M-CORD monitoring system (based on M-CORD 4.1)

헤드 노드 내에는 패브릭 스위치를 제어하기 위한 ONOS 컨트롤러들이 구동되며, M-CORD 내에는 M-CORD 전체의 서비스를 제어하기 위한 서비스 제어 평면인 XOS가 위치한다. 또한 ONOS 컨트롤러는 M-CORD 내의 패브릭 스위치들을 제어하기 위한 ONOS 어플리케이션을 사용한다. 마지막으로 VNF들을 배포하고 가상 네트워크 생성을 위한 Kubernetes (또는 Docker)가 위치한다. 컴퓨트 노드에서는 EPC 네트워크가 VNF 컨테이너로써 동작한다.

이러한 M-CORD 환경을 위한 모니터링 시스템은 크게 모니터링 에이전트 (monitoring agents), 모니터링 서비스 (monitoring service) 및 대시보드 (dashboard)로 구성된다. M-CORD EPC 네트워크 구성요소는 모바일 서비스를 제공하고, 모니터링 에이전트는 각 물리적 노드 및 컨테이너에 대한 리소스 사용량 정보 및 네트워크 통계를 수집한다. 모니터링 서비스는 에이전트가 수집한 데이터를 가져와서 시계열 (time-series) 데이터베이스에 저장한다. 데이터베이스에 저장된 시계열 데이터는 웹 UI 기반 대시보드를 통해 사용자에게 제공된다. M-CORD 모니터링 시스템의 각 구성요소의 역할 및 기능적 요구사항은 다음과 같다.

- 물리 노드 및 VNF들을 모니터링하는 모니터링 에이전트는 M-CORD 플랫폼의 컴퓨트 노드 안에서 구동된다. 모니터링 에이전트는 Node exporter, cAdvisor 및 Packet classifier 3가지로 구성된다. Node exporter는 물리적 서버의 리소스 사용량 및 네트워크 통계 정보를 모니터링하고 cAdvisor는 각 컨테이너의 메트릭을 모니터링한다. Node exporter와 cAdivor가 수집하는 정보는 CPU 사용량, 메모리 사용량, 디스크 I/O, 디스크 용량 (capacity), 네트워크 인터페이스의 TX/RX Throughput 등 다양한 정보들을 수집한다. Packet classifier는 VNF 컨테이너 사이의 가상 네트워크 인터페이스를 모니터링하는 Python 프로그램으로, EPC 네트워크에서 오가는 트래픽을 분류하고 각 UE가 송신하거나 수신하는 트래픽을 집계한다. 각 에이전트는 5초마다 리소스 사용량과 네트워크 통계 정보를 수집한다.

- 모니터링 서비스는 모니터링 에이전트에서 수집한 정보를 데이터베이스에 저장하고 웹 UI 기반 대시보드에 데이터를 제공한다. 모니터링 서비스는 모니터링 솔루션인 Prometheus와 시계열 데이터베이스인 InfluxDB로 구성된다. Prometheus는 주기적으로 각 모니터링 에이전트의 HTTP 엔드 포인트에 연결하여 에이전트가 수집한 정보를 가져온다. 가져온 정보는 Prometheus 서버 내의 시계열 데이터베이스 (TSDB)에 저장된다. 또한 우리는 InfluxDB를 사용하여 Packet Classifier가 트래픽 타입 및 각 UE에 대해 집계한 트래픽 통계 정보를 저장한다. InfluxDB는 각 UE가 송수신하는 EPC 네트워크 내부의 트래픽에 대해 프로토콜 유형 (TCP, UDP 등) 별로 분류 된 패킷 카운트 및 패킷 바

이트 수를 집계하여 저장하고 있다. 이러한 Prometheus와 InfluxDB는 대시보드와 상호 작용하여 모니터링 데이터를 사용자에게 제공한다.

- 웹 GUI는 시계열 차원의 분석을 위한 오픈 소스 소프트웨어 인 Grafana 대시보드를 사용한다. Grafana는 Prometheus 서버의 내부 데이터베이스에 저장된 실제 서버 및 EPC 네트워크 구성요소의 모니터링 정보와 InfluxDB에 저장된 네트워크 통계 정보를 그래프와 같은 시각화 형태로 관리자에게 제공한다. 또한 쿼리를 통해 관리자가 원하는 데이터를 수집하여 그래프, 표 및 계기와 같은 원하는 형식으로 시각화 할 수 있다.

이러한 모니터링 시스템을 통해 기대할 수 있는 서비스는 다음과 같다. 먼저 M-CORD를 구성하는 다양한 노드들의 전반적인 컴퓨팅 및 네트워킹 자원들을 시각화 할 수 있다. 이는 네트워크 관리자로 하여금 M-CORD의 상태를 쉽게 인지할 수 있도록 한다. 이를 확장하여, 만일 M-CORD 내부의 자원이 부족하거나 과도히 사용되는 경우 능동적으로 이 자원을 조절하여 확장성 (scalability)을 제공할 수 있는 장점이 있다. 또한, 전반적인 자원의 시각화뿐만 아니라, 현재 VNF들이 어떠한 트래픽을 송/수신하는지를 확인할 수 있다. 이를 통해 외부에서 정상적이지 않은 트래픽이 접근한다거나, 인증 받지 않은 사용자가 EPC 네트워크로 접근을 요구하는 등의 보안을 위한 트래픽 모니터링을 쉽게 할 수 있다.

4.2. Implementation

그림 8은 제안하는 모니터링 시스템의 동작 방식을 나타낸다. 모니터링 시스템은 크게 시스템 리소스 모니터링과 트래픽 모니터링으로 구성된다. 먼저, 시스템 리소스 모니터링은 Node exporter와 cAdvisor를 사용하여 Prometheus를 통해 컨테이너와 물리 서버에 대한 리소스 사용량을 수집한다. Node exporter는 컴퓨트 노드에서 Docker에서 데몬 컨테이너로 작동하며 물리 서버 (host OS)의 메트릭을 수집한다. cAdvisor는 Node exporter와 마찬가지로 데몬 컨테이너의 형태로 작동하고 Docker 엔진을 통해 실행되는 각 컨테이너의 메트릭을 수집한다. 두 에이전트가 메트릭을 수집하는 동안 Prometheus는 Service Discovery를 통해 모니터링 타겟 (Node exporter, cAdvisor)을 찾고, 각 대상의 HTTP 엔드포인트를 통해 액세스하여 수집된 메트릭을 주기적으로 가져온다. 에이전트에서 가져온 메트릭은 Prometheus 서버 내의 TSDB에 저장된다.

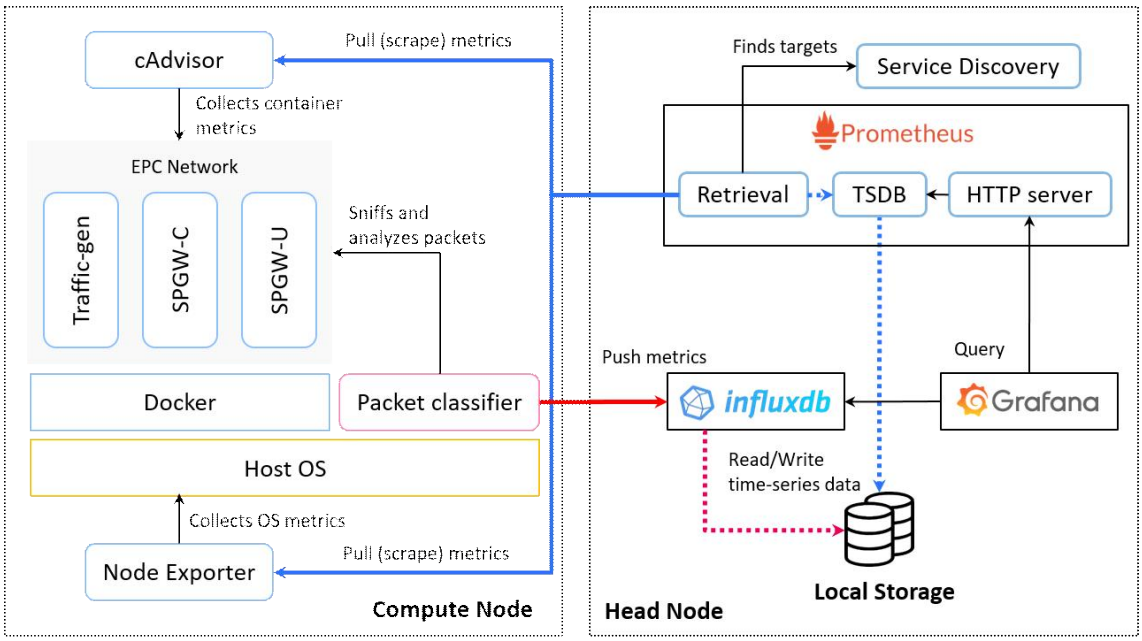

그림 8. Implementation architecture of M-CORD monitoring system

다음으로 트래픽 모니터링은 InfluxDB와 Python으로 작성된 Packet classifier 프로그램으로 구성된다. Packet classifier는 Scapy 라이브러리를 사용하여 EPC 네트워크의 내부 인터페이스를 모니터링하고 패킷을 스니핑한다. 스니핑된 패킷은 5-tuple (source IP, destination IP, protocol ID, source 포트 번호, destination 포트 번호)을 기반으로 패킷 수와 패킷 바이트를 집계하여 분석한다. 하지만 GTP 터널 안에서는 original 패킷이 GTP 터널 헤더에 캡슐화되기 때문에 원래 패킷의 5-tuple 정보가 추출된다. 또한 각 UE 별 트래픽을 모니터링하기 위해 패킷 카운트와 패킷 바이트

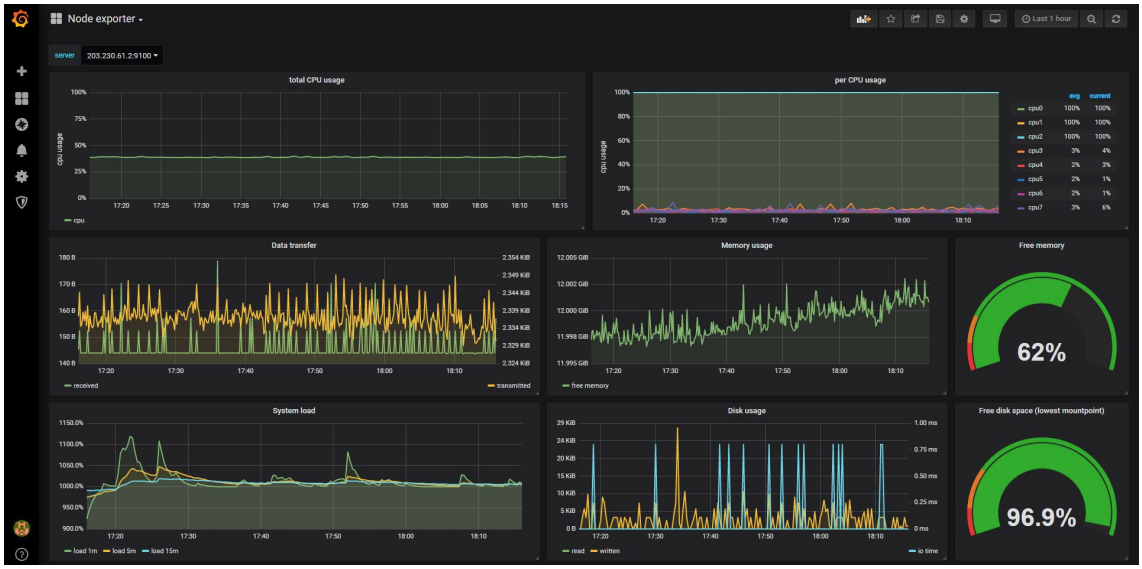

그림 9. Grafana dashboard of M-CORD monitoring system

수가 eNB와 SPGW-U (s1u 인터페이스) 간의 인터페이스에서 각 UE의 연결 마다 생성된 터널 ID 인 TEID를 기반으로 집계된다. 집계된 메트릭은 API를 사용하여 InfluxDB 서버로 푸시되고 시계열 데이터 형식으로 데이터베이스에 저장된다.

마지막으로 Grafana 대시보드는 InfluxDB 및 Prometheus에 저장된 정보를 웹 UI를 통해 사용자에게 제공하는데 사용된다. Grafana는 Prometheus 및 InfluxDB 서버에 각각 HTTP API를 통해 쿼리 (PromQL 및 InfluxQL)를 전송하여 필요한 모니터링 데이터를 수신하고, 이를 시각화하여 관리자에게 제공한다. 이를 통해 구현된 모니터링 시스템은 그림 9와 같다.

5. A Design of M-CORD Anomaly Detection

4절에서 제안하고 구현한 M-CORD 모니터링 시스템을 바탕으로 머신 러닝 (Machine Learning; ML) 학습 모델을 활용하여 이상 상태를 탐지하는 anomaly detection 서비스를 디자인 한다. Anomaly detection은 물리 서버 및 VNF의 시스템 리소스 및 네트워크 이상 상태를 실시간으로 감지하는 것으로, 전체적인 구조는 그림 10과 같다. 그림 10의 좌측과 같이 M-CORD 플랫폼은 모바일 서비스 및 다양한 서비스를 제공하고, 이를 모니터링 시스템을 통해 M-CORD 플랫폼에 위치한 각 물리 서버 및 VNF의 시스템 리소스 및 네트워크 상태를 모니터링한다. 모니터링되는 정보는 모니터링 시스템의 TSDB에 저장이 되어 anomaly detection 서비스에 활용된다.

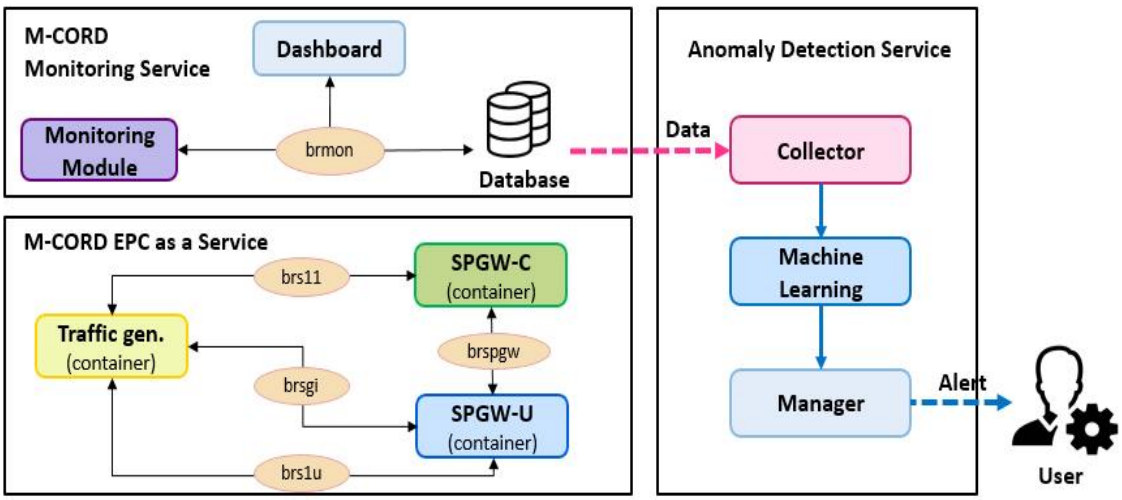

그림 10. Anomaly detection using M-CORD monitoring system

Anomaly detection 서비스는 크게 그림 9의 우측과 같은 3가지 프로세스를 통해 이루어진다. 먼저 모니터링 시스템을 통해 수집된 정보를 가져오는 Collector는 TSDB로부터 가져온 정보를 학습에 활용될 수 있도록 전처리 (Pre-processing)하는 과정을 거치고, 지도 학습 (Supervised Learning)을 이용하는 경우 그에 대한 레이블링 (labeling) 작업을 한다. 이때 학습에 활용되는 정보는 텍스트 파일 또는 csv 파일의 형태로 전처리 과정이 이루어진다. M-CORD anomaly detection 서비스를 제공하는 각 구성요소의 역할 및 기능적 요구사항은 다음과 같다.

- Collector는 M-CORD 모니터링 시스템을 통해 데이터베이스에 저장된 정보를 가져온다. Collector로 전달된 모니터링 정보는 머신러닝 및 딥러닝 알고리즘 학습에 활용될 수 있도록 전처리를 한다. 전처리 과정은 CPU 및 메모리 사용량, 디스크 I/O, 트래픽 로드 등과 같은 모니터링 정보들을 시간적 순서(historical)로 정리한다. 그 후, 실제 정상적이지 않은 동작이 있었거나 fault injection을 통해 생성된 이상 상태의 데이터를 레이블링하는 작업을 거친다. 이렇게 전처리된 데이터셋은 머신러닝/딥러닝 알고리즘 학습에 활용된다.

- 다음으로 전처리가 끝난 데이터셋을 이용한 학습 모델 생성은 머신러닝 또는 딥러닝 알고리즘을 이용한다. 학습은 구글의 TensorFlow를 이용하거나 다양한 머신러닝 알고리즘을 제공하는 Scikit-learn 라이브러리 및 AutoML을 활용하여 간단한 머신러닝 알고리즘인 SVM, Decision Tree 분류 모델부터 RNN, LSTM 과 같은 딥러닝 및 GBM, XGBoost과 같은 앙상블 러닝 (ensemble learning) 등의 정확도를 비교하여 최적의 학습 모델을 찾는다.

- 마지막으로 Manager 모듈은 학습을 통해 생성된 anomaly detection 모델을 M-CORD 플랫폼 환경에 적용시키는 단계이다. 머신러닝 및 딥러닝 알고리즘을 통해 생성된 anomaly detection 모델을 바탕으로 Manager는 REST API를 통해 M-CORD 플랫폼에서 동작하는 물리 머신 및 VNF들에 대한 현재 상태 정보 (리소스 사용량, 트래픽 로드 등)를 가져와 이를 바탕으로 물리 머신 및 VNF들의 리소스 사용량 및 트래픽 사용량이 적절한지를 판단한다. 만약 CPU 및 메모리 등 리소스가 너무 많이 사용되고 있거나 트래픽이 과다하게 몰리는 경우 등과 같은 비정상적인 상태를 감지할 경우 사용자 (관리자)에게 이를 알려주는 역할을 한다.

이러한 Anomaly detection 서비스는 서버 및 컨테이너의 현재 상태에 관련된 정보를 판단하지만 나아가 이상 상황이 발생하기 전에 미리 예측하여 이상 상황을 사전에 감지 (proactive)하는 모델을 만드는 것도 가능할 것으로 기대된다.

6. Conclusion

본 기술문서에서는 CORD와 M-CORD의 등장배경 및 개념을 설명하고, M-CORD 플랫폼 환경에서 현재 부재하는 모니터링 기능의 지원 방안으로 M-CORD 모니터링 시스템을 디자인 및 구현하고, 그리고 이러한 모니터링 시스템을 활용한 anomaly detection 서비스에 대해 디자인을 제안하였다. 모니터링 시스템을 활용하면 기존 M-CORD 플랫폼에서 사용 중인 다양한 각종 물리 노드들의 CPU 및 메모리 사용량 등은 물론, M-CORD에서 동작 중인 각 VNF들의 리소스 사용량 및 트래픽 로드 (traffic load)에 대해 모니터링할 수 있다. 또한 모니터링 정보를 활용한 anomaly detection 서비스는 머신러닝 알고리즘을 통해 학습한 모델을 통해 각 물리 노드 및 VNF들의 자원 사용량 및 네트워크 상태의 이상 상태를 감지하여 M-CORD 플랫폼을 운용하는데 도움을 줄 것으로 기대된다. 이를 바탕으로 우리는 M-CORD 모니터링 시스템을 이용한 anomaly detection 서비스를 구현할 예정이다.

K-ONE 기술 문서

- 이 문서의 기술적인 내용은 프로젝트의 진행과 함께 별도의 예고 없이 변경될 수 있습니다.

- 본 문서와 관련된 문의 사항은 아래의 정보를 참조하시길 바랍니다.
  (Homepage: http://opennetworking.kr/projects/k-one-collaboration-project/wiki, E-mail: k1@opennetworking.kr)

작성기관: K-ONE Consortium
작성년월: 2019/07