OpenTelemetry와 Elastic APM의 관측성(Observability) 기능 비교 연구

지동혁*, 최상훈[1], 박기웅†

세종대학교 SysCore Lab. (대학원생, 연구 교수[1])

**세종대학교 정보보호학과 (교수†)

Comparative Analysis of Observability Capabilities: OpenTelemetry and Elastic APM

Dong-Hyeok Ji*, Sang-Hoon Choi[1], Ki-Woong Park†

* Syscore Lab., Sejong University
**Dept. of Computer and Information Security, Sejong University
(Graduate Student*, Research Professor[1], Professor†)

요 약

최근 시스템의 복잡성이 증가함에 따라, 관측성 도구를 활용한 성능 및 보안 모니터링의 중요성이 커지고 있다. 본 연구는 OpenTelemetry(이하 OTel)와 Elastic APM을 비교하여, 각 도구의 구조와 기능, 적용 환경에 따른 특성을 분석하였다. 웹 환경에서 반복 요청을 발생시켜 메트릭 및 트레이스를 수집하고 시각화 결과를 비교한 결과, OTel은 마이크로서비스에 적합한 유연성과 확장성을, Elastic APM은 단일 서비스에 적합한 간편성과 직관성을 보였다. 본 연구는 관측성 도구 선택 기준을 제시하고, 향후 보안 운영 적용 가능성을 논의한다.

I. 서론

최근 기업들은 시스템의 유연성과 비용 효율성을 높이기 위해 클라우드 환경으로 마이그레이션을 진행하고 있다. 동시에 IT 인프라는 모놀리식 아키텍처에서 마이크로서비스 아키텍처(MSA)로의 전환이 가속화되고 있다. 이 같은 변화는 인프라가 복잡해지고 관리 대상이 분산되면서 서비스의 상태를 종합적으로 파악하기 위한 새로운 요구를 만들어냈다 [1][2].

특히 기존의 단순한 모니터링 방식만으로는 성능 저하의 원인을 효과적으로 분석하기 어렵다. 이에 따라 관측가능성(Observability) 개념이 주목받기 시작했으며, 단순한 모니터링을 넘어 보안 관점에서의 관측성 확보 역시 점차 중요성이 커지고 있다 [3].

이러한 배경을 바탕으로, 본 연구에서는 현대 소프트웨어 환경에서 널리 활용되는 대표적인 관측성 도구인 OTel과 Elastic APM을 선정하여 그 구조와 기능을 비교 분석하였다.

두 도구는 각각 마이크로서비스 기반 환경과 단일 서비스환경에서 주로 사용되며, 관측성 확보라는 목표를 가지고 있다 [4]. 본 연구는 웹 경로에 대해 다수의 요청을 보내는 방식의 부하 테스트를 설계하였다. 이는 DoS 공격과 유사한 패턴으로, 시스템의 응답 지연 등의 가용성 침해를 유발할 수 있다. 이를 바탕으로 각 도구가 수집한 메트릭과 트레이스를 어떤 방식으로 시각화하는지를 비교하고, 도구들의 장단점을 분석하는 실험을 설계하였다. 본 논문의 2장에서 기술 배경을 기술하고, 3장에서는 두 도구에 대한 관측 기능 평가 및 비교, 4장에서는 결론 및 향후 연구 방향을 제시한다.

† 교신저자: 박기웅 (세종대학교 정보보호학과 교수)

본 논문은 과학기술정보통신부의 재원으로 실감콘텐츠핵심기술개발(Project No. RS-2023-00228996,40%), 한국연구재단(NRF) 중견후속연구사업(Project No. RS-2023-00208460, 30%), 국방ICT융합연구(Project No. 2022-11220701, 30%)의 지원을 받아 수행된 연구임.

II. 기술 배경

2.1 관측성(Observability) 핵심 요소

메트릭은 시스템이나 애플리케이션의 상태를 수치상으로 표현한 데이터이다. CPU 사용량, 메모리 사용량 등과 같은 실시간 성능 지표를 수집하고, 주기적으로 수치 변화를 기록한다. 로그는 시스템에서 발생한 이벤트를 시간순으로 기록한 데이터이다. 오류 메시지, 정보성 메시지 등이 포함되며, 문제 발생 시 원인을 분석하는 데 중요한 역할을 한다. 트레이스는 하나의 요청이 시스템 내부를 어떻게 이동했는지를 기록한다. 이는 병목 현상이나 장애 지점을 식별하는 데 필수적이다. 참고로 본 연구에서 관측성 도구는 로그 데이터를 수집하더라도 보안 공격 탐지를 수행하지 않으므로 로그 데이터는 연구 범위에서 제외한다 [1][4].

2.2 OTel 개요

OTel은 클라우드 네이티브 환경을 위한 오픈소스 관측성 프레임워크이다. OTel은 기존의 OpenTracing과 OpenCensus 프로젝트를 통합하여 만들어졌으며, 다양한 백엔드 시스템으로 데이터를 전송할 수 있도록 설계되었다 [4]. 본 연구에서는 OTel Collector, Jaeger, Prometheus, Grafana를 사용한다. Collector는 OTel SDK가 생성한 메트릭, 로그, 트레이스를 수집하고 외부 도구에 전송하는 중간 허브 역할을 한다. Collector는 파이프라인 기반 구조로 되어 있으며, 데이터 수신, 처리, 전송 단계로 구성된다. Jaeger는 수집된 트레이스를 데이터 기반으로 병목 현상 및 지연 구간을 파악할 수 있으며 가벼운 트레이싱 환경을 구축하고 싶을 때 사용한다. Tempo는 Jaeger와 같은 분산 추적을 위한 도구이고, 복잡한 트레이싱 환경에서 쓰인다. Prometheus는 시계열 메트릭을 수집하고 저장하는 오픈소스 모니터링 시스템이다. 애플리케이션 내부 상태를 수치화해서 보게 해주고 상태를 이해할 수 있는 기반을 제공한다. Grafana는 수집된 모든 데이터를 실시간으로 대시보드와 그래프로 시각화해 주는 대시보드 도구이다 [5][6].

2.3 Elastic APM 개요

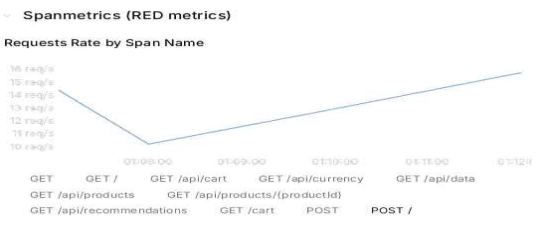
Elastic APM은 Elastic Stack의 핵심 구성 요소로, 애플리케이션의 성능과 상태를 모니터링할 수 있도록 지원하는 오픈소스 APM 솔루션이다 [8]. 본 연구에서는 APM Agent, Elasticsearch, Kibana를 사용한다. Agent는 애플리케이션 내부에 설치되어 실행되며, 트랜잭션, 스팬, 메트릭 데이터를 실시간으로 수집하는 컴포넌트이다. Elasticsearch는 대용량의 데이터를 실시간으로 검색하고 분석, 저장할 수 있는 오픈소스 검색 엔진이다 [9]. Kibana는 Elasticsearch에 저장된 데이터를 시각화하고 분석할 수 있게 해주는 오픈소스 대시보드 도구이다. Elastic APM은 단일 서비스 기반으로 동작하기 때문에 분산 추적 단위인 트레이스보다는 트렌젝션 단위로 분석을 진행한다 [10].

III. OTel과 Elastic APM에 대한 관측 기능 평가 및 비교

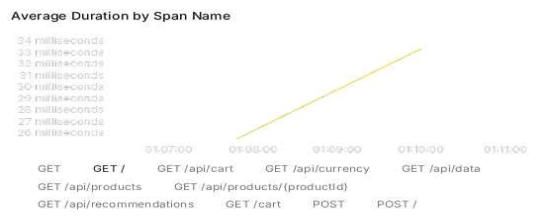
3.1 실험 구성 및 전제

실험에서 OTel 환경은 3,000회의 요청을, Elastic APM 환경은 2,000회의 요청을 curl 명령어를 통해 반복적으로 전송하였다. 이는 시스템 자원 과부하를 피하면서 실험 결과를 시각적으로 분석하기 위한 목적으로 설정한 값이다.

3.2 메트릭 기능 및 시각화



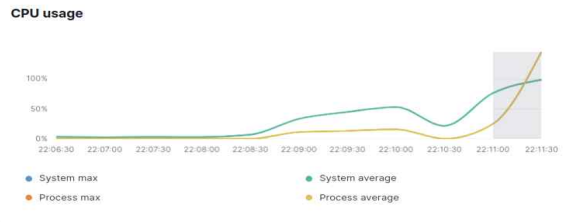
(그림 1) 스팬 시간대별 요청 수



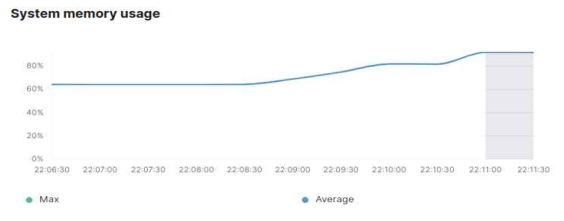
(그림 2) 스팬 시간대별 평균 응답 시간

[그림 1]은 스팬의 요청률 변화를 나타낸다. 이때 POST 스팬의 요청 수는 실험 시점인

01:08 이후 급격히 상승하여 16 req/s까지 도달하였다. [그림 2]는 스팬의 평균 응답 시간을 나타내며, GET의 평균 응답 시간이 약 26ms에서 33ms까지 증가한 것을 확인할 수 있다.



(그림 3) 시스템, 프로세스 CPU 평균 사용률

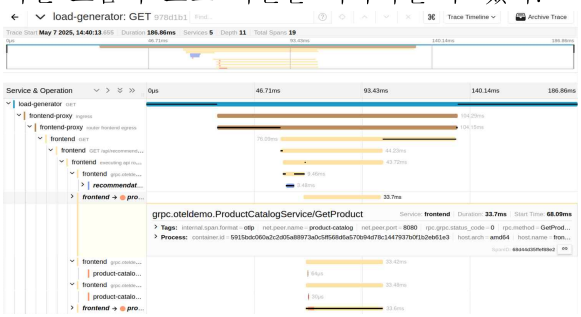


(그림 4) 시스템 메모리 평균 사용률

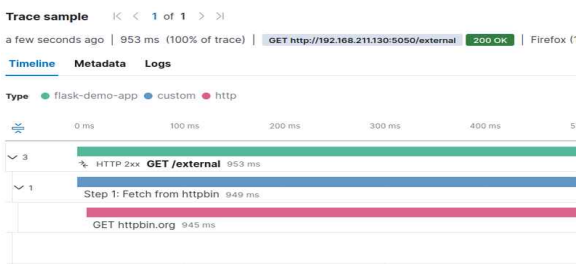
[그림 3]은 CPU 사용률의 변화를 나타낸다. 요청이 집중되면서 프로세스 평균은 100% 이상 상승하였고, 시스템 평균 또한 80% 이상으로 상승하였다. [그림 4]는 시스템 메모리 사용률을 보여주며, 평균 사용률이 약 63%에서 83%로 상승하였다.

3.3 트레이스 시각화

트레이스 비교는 요청 흐름을 각 도구가 어떻게 표현하는지를 중심으로 수행하였다. [그림 5]는 요청이 마이크로서비스를 거치며 생성된 스팬들을 계층적으로 표현하며, 사용자가 이용한 서비스 흐름과 지연 시간도 확인할 수 있다. [그림 6]의 Elastic APM 또한 계층 구조 기반의 타임라인을 제공하며, 애플리케이션 내 트렌젝션 흐름과 소요 시간을 시각화할 수 있다.



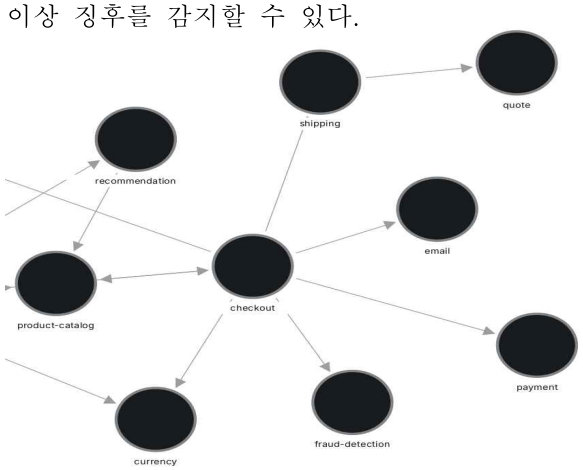
(그림 5) 트레이서 흐름과 지연 시간



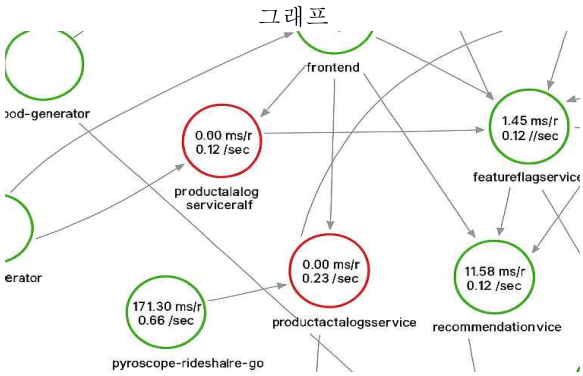
(그림 6) 트렌젝션 흐름과 지연 시간

3.4 호출 의존성

OTel에서는 [그림 7]과 같이 서비스 간 호출 관계를 시각적으로 표현하는 호출 의존성 기능을 제공한다. 본 실험에서는 단순한 마이크로서비스 환경을 구성했기 때문에 간단하게 보이지만, 대규모 환경에서는 이 호출 그래프가 복잡하게 확장될 것으로 예상된다. 특히 [그림 8]처럼 Tempo를 활용하면 각 서비스 노드 내에 평균 응답시간, 초당 요청 수 등의 지표를 함께 시각화할 수 있어, 특정 구간에서 발생하는 이상 징후를 감지할 수 있다.



(그림 7) Jaeger를 활용한 서비스 간 호출 관계 그래프



(그림 8) Tempo를 활용한 서비스 간 호출 관계 그래프

3.5 기능 비교 결과

항목	OTel	Elastic APM
시각화 정밀도	매우 높음	비교적 제한적
벤더 종속성	없음	있음
보안 탐지 기능	없음	없음
대시보드 구성	매우 자유로움	제한적
지원 프레임워크	다양한 언어 및 프레임워크 지원	Elastic이 제공하는 주요 언어 중심

(표 1) OTel와 Elastic APM 비교

IV.결론 및 향후 연구

본 비교 실험을 통해 OTel은 다양한 프레임워크를 지원하며 대시보드 구성의 자유도가 높고 마이크로 서비스환경에서 유연하게 활용이 가능하다는 장점이 있다. 반면 보안 위협을 탐지하는 기능은 제공하지 않아 다른 도구에 의존해야 하는 단점이 있다. Elastic APM은 트렌젝션을 중심으로 트레이서를 파악할 수 있어서 성능 모니터링엔 매우 특화되어 있다. 하지만 대시보드 활용도가 자유롭지 않다는 점과 보안 탐지 기능이 없다는 단점이 있다.

향후 연구에서는 다양한 그래프를 기반으로 실험 환경을 설계하여, OTel의 성능을 평가할 예정이다. 또한 호출 의존성 그래프와 보안 탐지 도구를 결합하여, 침해사고 발생 가능성이 높은 구간이나 병목 현상을 사전에 탐지할 수 있는 관측성 기반 보안 운영 방안을 제시하고자 한다.

[참고문헌]

[1] S. Chinamanagonda, “Observability in microservices architectures: Advanced observability tools for microservices environments,” MZ Comput. J., vol. 3, no. 1, pp. 1 - 22, 2022.

[2] S.-P. Ma, C.-Y. Fan, Y. Chuang, W.-T. Lee, S.-J. Lee, and N.-L. Hsueh, “Using service dependency graph to analyze and test microservices,” Proc. IEEE COMPSAC, pp. 81 - 86, 2018. doi: 10.1109/COMPSAC.2018.10207.

[3] M. C. Borges et al., "Informed and assessable observability design decisions in cloud-native microservice applications," Proc. IEEE ICSA, pp. 69 - 78, 2024. DOI: 10.1109/ICSA59870.2024.00015.

[4] U. Faseeha et al., “Observability in Microservices: Frameworks, Challenges, and Deployment,” IEEE Access, vol. 13, pp. 72011 - 72026, 2025. doi: 10.1109/ACCESS.2025.3562125

[5] G. Y. Kusuma and U. Y. Oktiawati, “Application performance monitoring using OpenTelemetry and Grafana Stack,” J. Internet Softw. Eng., vol. 3, no. 1, pp. 26 - 35, Nov. 2022. E-ISSN: 2797-9016.

[6] S. S. P. Sreemathy, V. Kumar C, and S. P. Priyadharshini, “Application monitoring and telemetry analytics,” Proc. ICCPCT, pp. 1559 - 1565, 2024. doi: 10.1109/ICCPCT61902.2024.10673415.

[7] A. Thakur and M. B. Chandak, “A review on OpenTelemetry and HTTP implementation,” Int. J. Health Sci. vol. 6, no. S2, pp. 15013 - 15023, Jun. 2022.

[8] I.Khokhriakov, V. Mazalova, and O. Merkulova, “Improving observability of SCADA systems using Elastic APM,” in Proc. ICALEPCS, 2023, pp. 1016 - 1021. doi: 10.18429/JACoW-ICALEPCS2023-WE3BCO04.

[9] T. Gatsi, X. P. Baloyi, J. L. Lekganyane, and R. L. Schwartz, “ELK stack deployment with Ansible,” Proc. ICALEPCS, pp. 1411 - 1414, 2023. doi: 10.18429/JACoW-ICALEPCS2023-THPDP047.

[10] M. Rohde et al., "Using an Elastic Stack as a base for logging and evaluation of public displays," Mensch und Computer Workshop,2023. DOI: 10.18420/muc2023-mci-ws13-303.