

실습 과제 2 보고서

2021320308 박한준

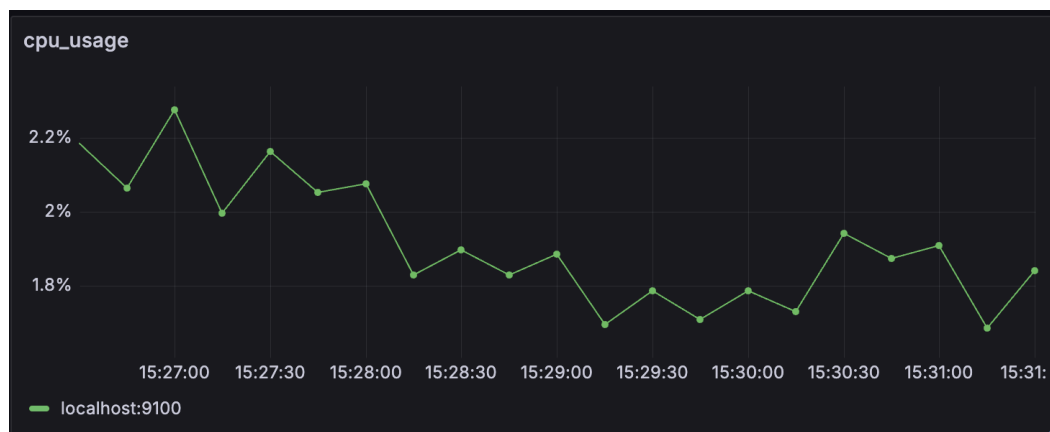
1. 그래파나 대시보드 (유휴 상태) & 간단 설명

- CPU usage (% , Time range=1m)

- 쿼리

```
100 * (sum by (instance) (rate(node_cpu_seconds_total{mode!="idle"}[1m])))  
/ (sum by (instance) (rate(node_cpu_seconds_total[1m])))
```

- 대시보드



- 설명

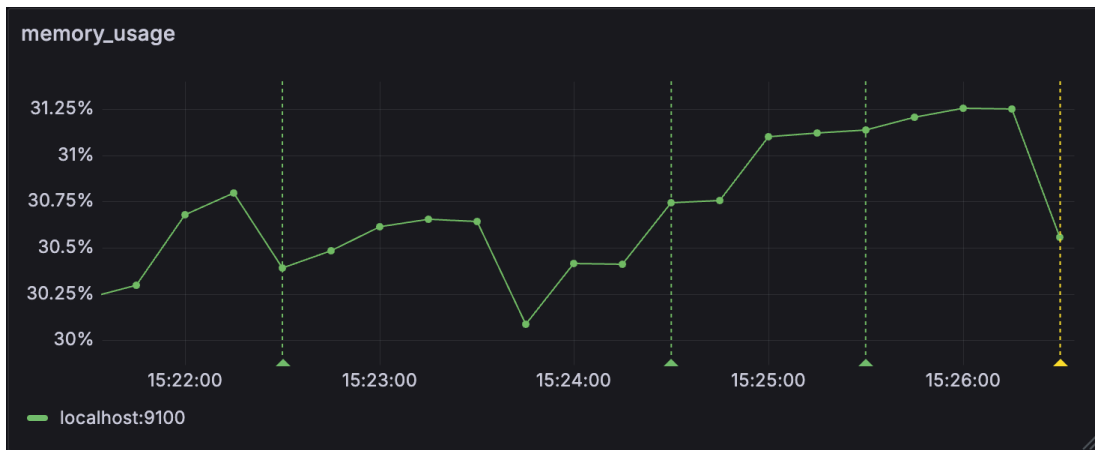
쿼리는 CPU의 유휴 모드를 제외한 누적 사용 시간을 나타낸다. 1분 동안의 CPU 사용 비율을 계산해 퍼센트로 표시하며, 전체 CPU 시간 대비 실제 사용 중인 비율을 보여준다. 이를 통해 서버의 실시간 부하 정도를 확인할 수 있다. 그래프는 OS를 유지하기 위한 최소한의 사용으로 2% 내외로 사용되고 있음을 알 수 있다.

- Memory usage (%)

- 쿼리

```
100 * (1 - (sum by (instance) (node_memory_MemAvailable_bytes)  
/ sum by (instance) (node_memory_MemTotal_bytes)))
```

- 대시보드



- 설명

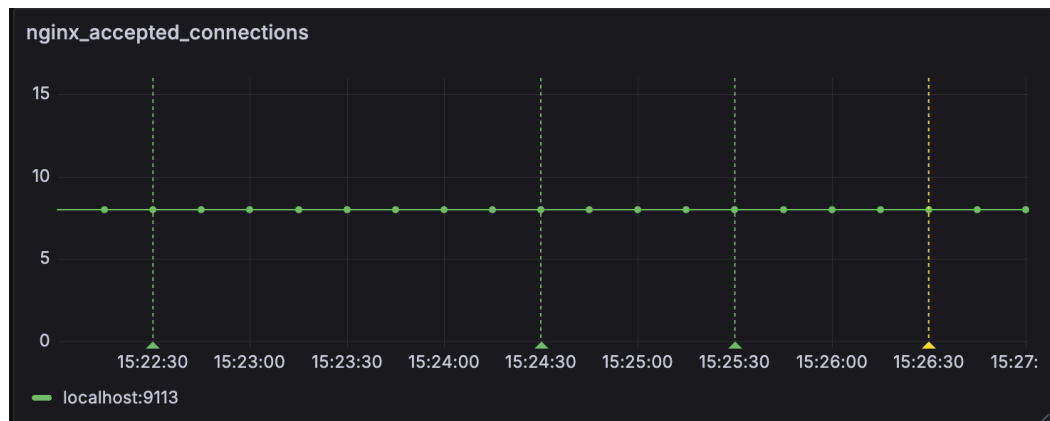
쿼리는 전체 메모리 중 사용 가능한 메모리를 제외한 비율을 계산해 현재 메모리 사용량을 퍼센트로 표시한다. `node_memory_MemAvailable_bytes` 와 `node_memory_MemTotal_bytes` 값을 이용해 실질적인 사용률을 구하며, 캐시와 버퍼 영역까지 포함해 실제 시스템 점유 상태를 반영한다. 그래프는 약 30% 내외로 안정적인 메모리 사용량을 보여주며, 현재 서버가 여유 있는 상태임을 알 수 있다.

- Nginx accepted connections

- 쿼리

```
sum by (instance) (nginx_connections_accepted)
```

- 대시보드



- 설명

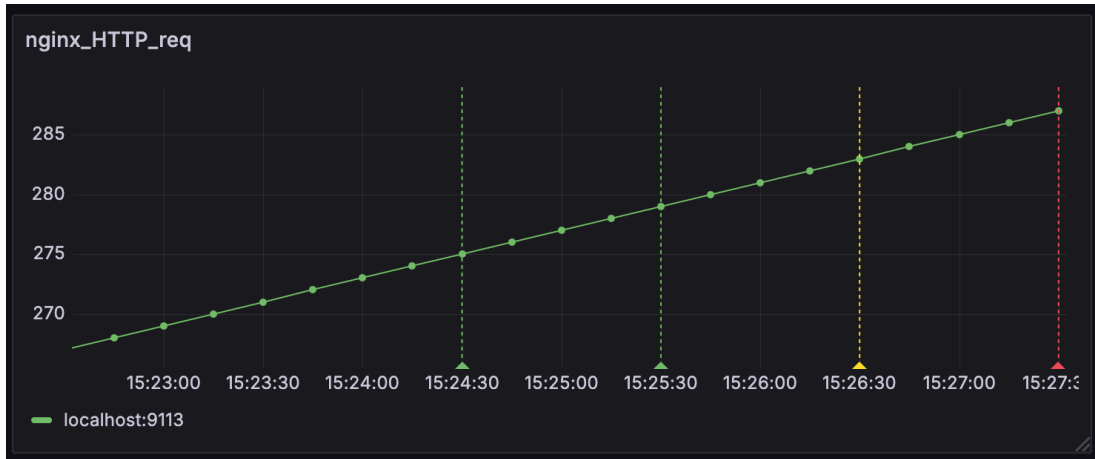
쿼리는 Nginx에서 수락된 연결 수를 인스턴스별로 합산해 표시한다. `nginx_connections_accepted` 메트릭은 서버가 처리한 누적 연결 수를 나타내며, 이를 통해 접속 요청이 얼마나 발생했는지 확인할 수 있다. 그래프는 일정하게 유지되고 있어 표시된 시간동안 새로운 연결 요청이 거의 없는 안정적인 상태임을 보여준다.

- Nginx HTTP Requests

- 쿼리

```
sum by (instance, method) (nginx_http_requests_total)
```

- 대시보드



◦ 설명

쿼리는 Nginx에서 처리된 HTTP 요청의 누적 횟수를 인스턴스와 요청 메서드별로 합산해 표시한다.

`nginx_http_requests_total` 메트릭은 서버가 처리한 전체 요청 수를 나타내며, 이를 통해 요청이 지속적으로 증가하는 추세를 확인할 수 있다. 그래프는 일정한 기울기로 증가하고 있어 서버가 안정적으로 HTTP 요청을 처리하고 있음을 보여준다.

2. ApacheBench 부하 테스트

- Small load: `ab -n 2000 -c 100 http://localhost/`



- Medium load: `ab -n 10000 -c 500 http://localhost/`



- Large load: `ab -n 100000 -c 1000 http://localhost/`



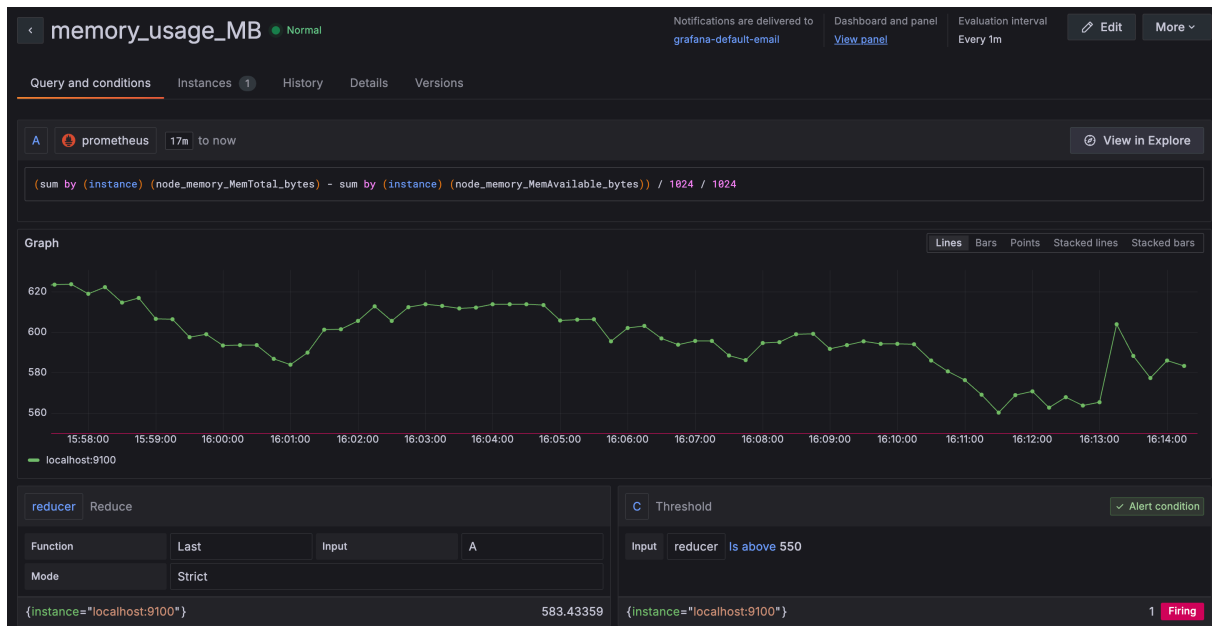
• 설명

아파치 벤치로 부하 테스트를 실행한 결과들이다. http request가 증가하는 패턴과 동일하게 nginx accepted connection도 증가하는 것을 볼 수 있다. 즉, 부하 테스트의 요청들이 성공적으로 수락되었다.

cpu 사용량은 **small load**에 대해서는 큰 변동이 없었다. **medium load**에서는 요청이 오는 타이밍에 5% 대로 상승했으나 요청이 더 이상 들어오지 않는 시점에도 한 번 5%로 상승한 모습이 포착된다. 이는 alert를 보내는 작업 때문일 수 있다. 결과적으로 medium load까지도 cpu 사용량이 크게 증가하지는 않는 모습을 확인할 수 있었다. **large load**에서는 request 와 accepted connection도 한 번에 고점을 찍는 것이 아니고 한 차례 보내고 잠시 뒤 나머지 요청을 보냈다. cpu 사용량은 25% 가까이 찍은 모습을 볼 수 있다.

모든 테스트에서 메모리 사용량은 큰 변동 없이 31% 내외에서 안정적인 모습을 보였는데, 이는 아파치 벤치의 부하 테스트가 cpu bound 작업임을 알 수 있다.

3. alert

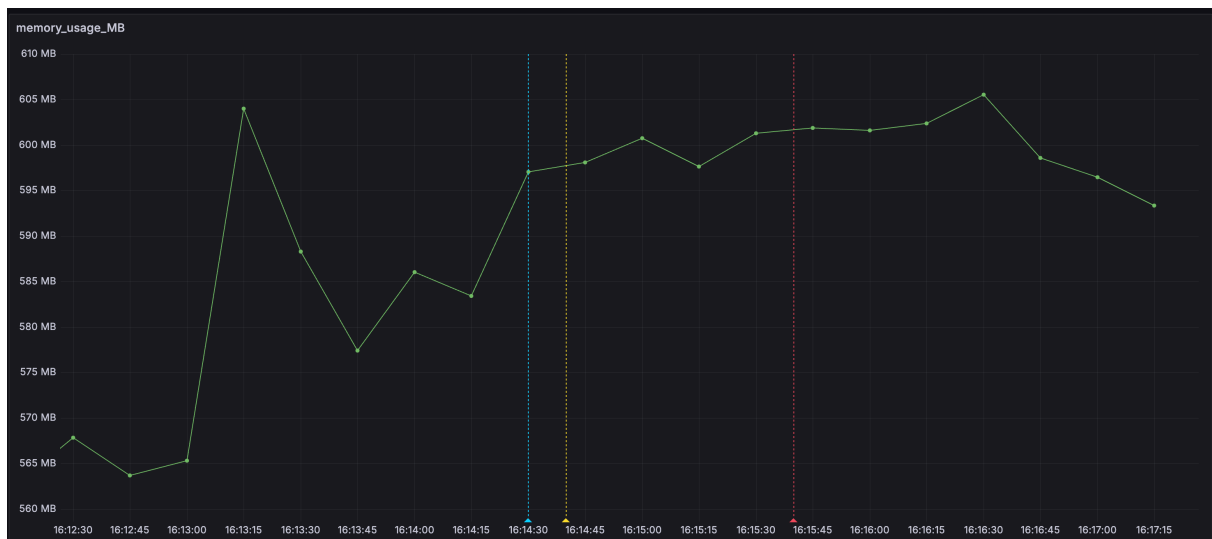


1 rule **1 firing**

Grafana-managed [Export rules](#) | [+ New recording rule](#)

assign2 > assign2 1 firing | [1m](#) | [View](#) | [Edit](#) | [More](#)

State	Name	Health	Summary	Next evaluation	Actions
Firing for 10s	memory_usage_MB	ok		in a minute	View Edit More



assign2 › memory_usage_MB

🔥 1 firing instances

Firing
memory_usage_MB
View alert

Values

C=1 reducer=601.2734375

Labels

alertname	memory_usage_MB
grafana_folder	assign2
instance	localhost:9100

Silence
View dashboard
View panel

Observed 30s before this notification was delivered, at 2025-11-05 15:15:40 +0800 CST

• 설명

memory_usage_MB 쿼리는 node_memory_MemTotal_bytes 에서 node_memory_MemAvailable_bytes 를 뺀 값(즉, 실제 사용 중인 메모리)을 MB 단위로 계산하는 쿼리이다. 알람 조건은 이 쿼리의 마지막(Last) 값이 550MB를 초과할 때(Is above 550) 'Firing' 상태가 되도록 설정했다.

대시보드 그래프는 알람 설정을 완료한 16:14:00 이후 시점에, 메모리 사용량이 550MB 이상이어서 상태가 **OK (파란색) → Pending (노란색) → Firing (빨간색)** 순으로 전환된 것을 볼 수 있다.

이후 Grafana가 알람 조건을 충족했다고 판단해 'Firing' 상태로 변경되었고, 알람이 이메일로 발송되었다. 즉, 설정된 조건이 정상적으로 감지되고, 알람이 트리거되어 전송까지 완료된 상황을 보여준다.