elastic

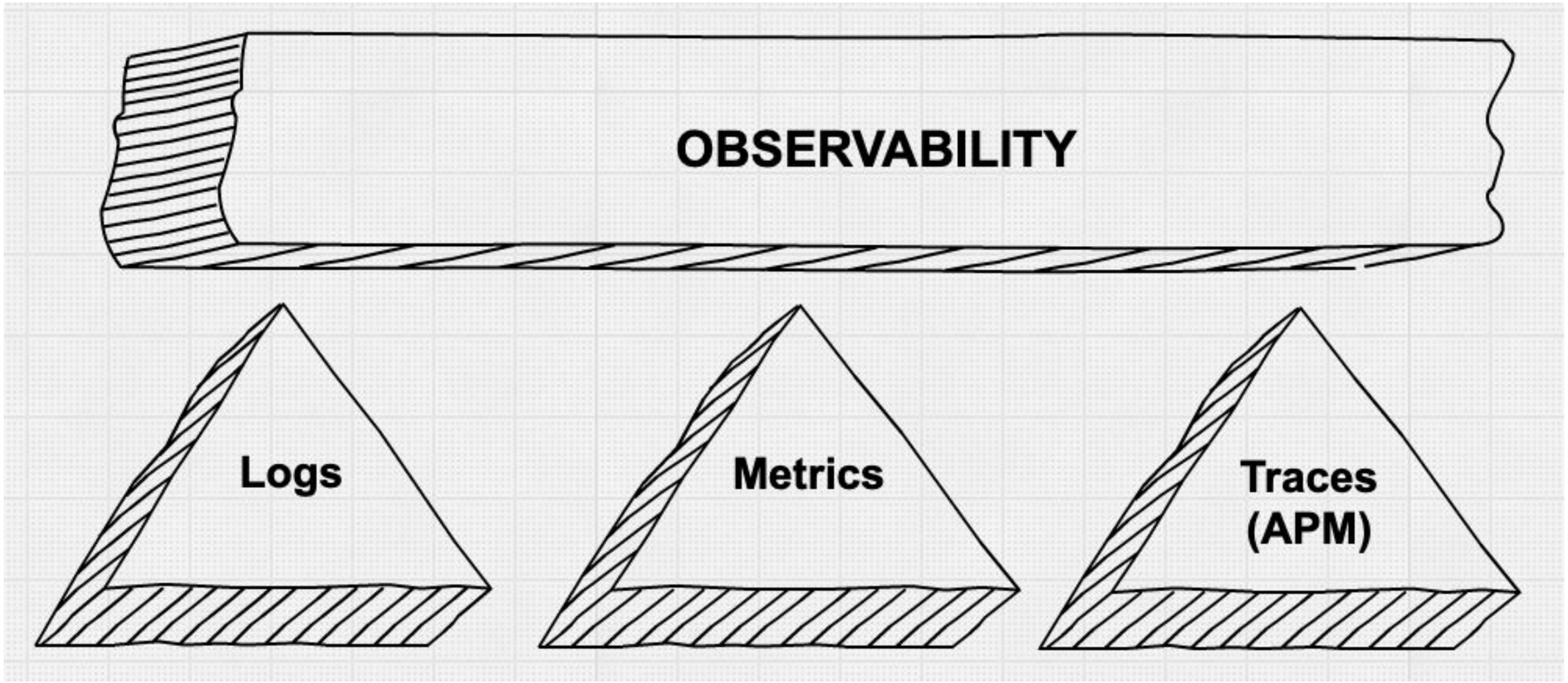
마이크로서비스 아키텍처 에서 **Elastic Observability** 활용

2021-05-20

김종민 (Jongmin Kim)
Developer 🥑 @Elastic

Observability

통합모니터링의 세 가지 데이터 타입



elastic

로그

특정 이벤트가 발생할 때마다 생성되는 데이터

```
64.242.88.10 - - [07/Mar/2017:16:10:02 -0800] "GET /mailman/listinfo/hsdivision HTTP/1.1" 200 6291
64.242.88.10 - - [07/Mar/2017:16:11:58 -0800] "POST /twiki/bin/view/TWiki/WikiSyntax HTTP/1.1" 404 7352
64.242.88.10 - - [07/Mar/2017:16:20:55 -0800] "GET /twiki/bin/view/Main/DCCAndPostFix HTTP/1.1" 200 5253
```

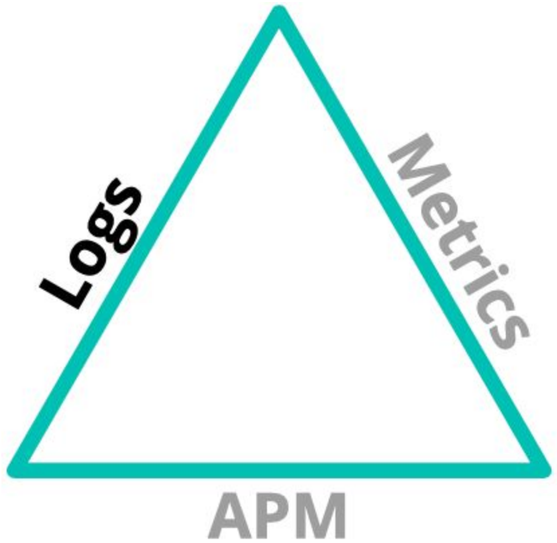
각 로그는 '어떤' 일이 일어났는지 기록

장점:

- 주로 애플리케이션 또는 서비스가 돌아가는 호스트에 남음
- 사람이 읽을 수 있고 커스터마이징이 가능

단점:

- 애플리케이션 수준이 아닌 컴포넌트 수준에서 기록됨
- 출력하지 않으면 기록이 남지 않음
- 포맷이 중요함



elastic

메트릭

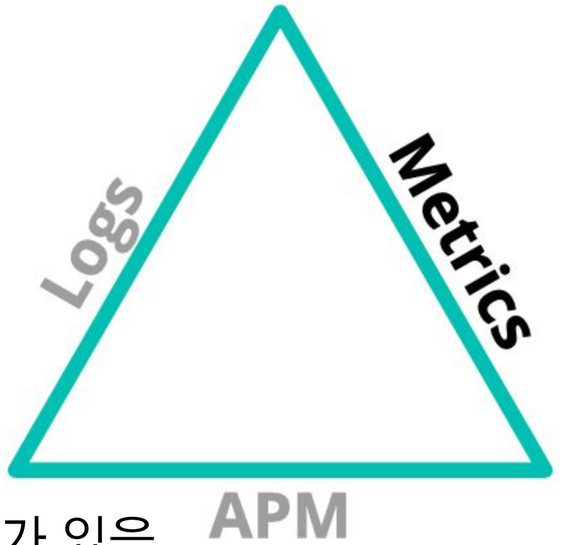
주기적으로 특정 값을 측정한 데이터

```
07/Mar/2017 16:10:00  all 2.58   0.00   0.70   1.12   0.05   95.55   server1   containerX   regionA
07/Mar/2017 16:20:00  all 2.56   0.00   0.69   1.05   0.04   95.66   server2   containerY   regionB
07/Mar/2017 16:30:00  all 2.64   0.00   0.65   1.15   0.05   95.50   server2   containerZ   regionC
```

예) 매 10초 마다 CPU 사용률을 측정해서 기록 함

장점:

- 트랜드와 이력을 보여줌
- 특정 사건이나 이상치를 잡기 위해 간단한 경고를 만들기 용이함



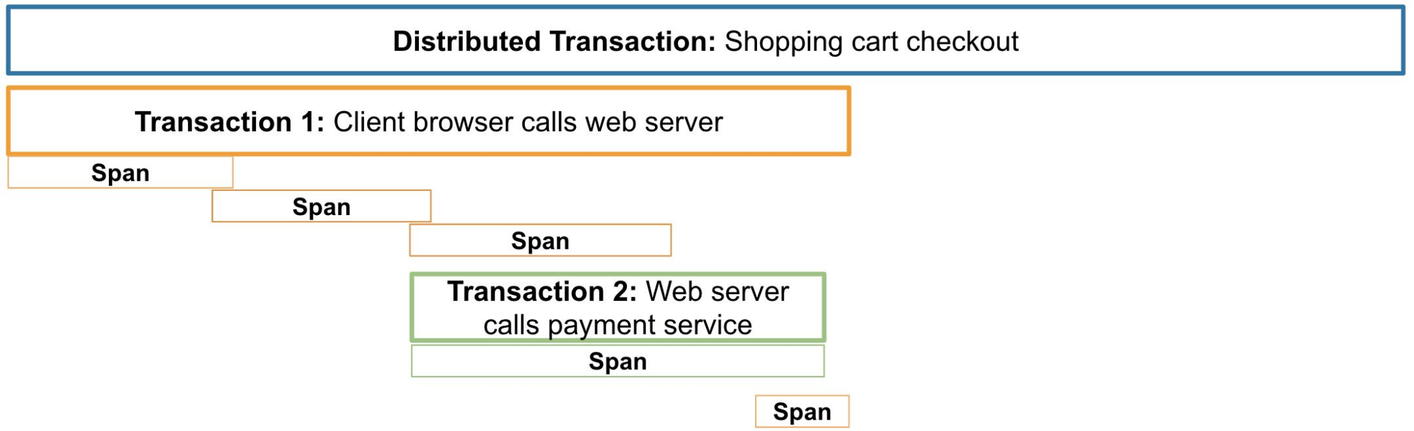
단점:

- 측정 주기 안에 있는 이상치는 평균에 묻혀버릴 수 있음
- 호스트 또는 네트워크에 기반하지만 컨테이너 환경에서는 총계값이 왜곡될 소지가 있음

elastic

트레이스

애플리케이션 코드 안에서 일어나는 활동을 기록한 데이터



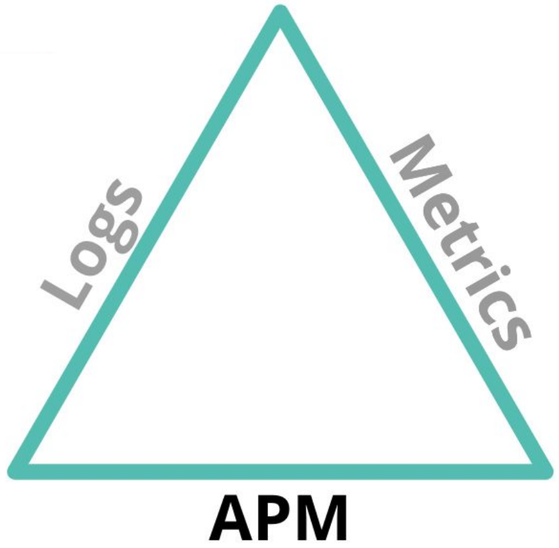
각 트랜잭션 마다 관련된 모든 컴포넌트와 그 세부 실행 시간을 추적

장점:

- 근본 원인 분석을 가능케 하는 매우 상세한 정보 제공

단점:

- 데이터 양이 매우 많고 내용이 장황함
- 애플리케이션 코드를 인스트루먼트(instrument) 할 필요가 있음

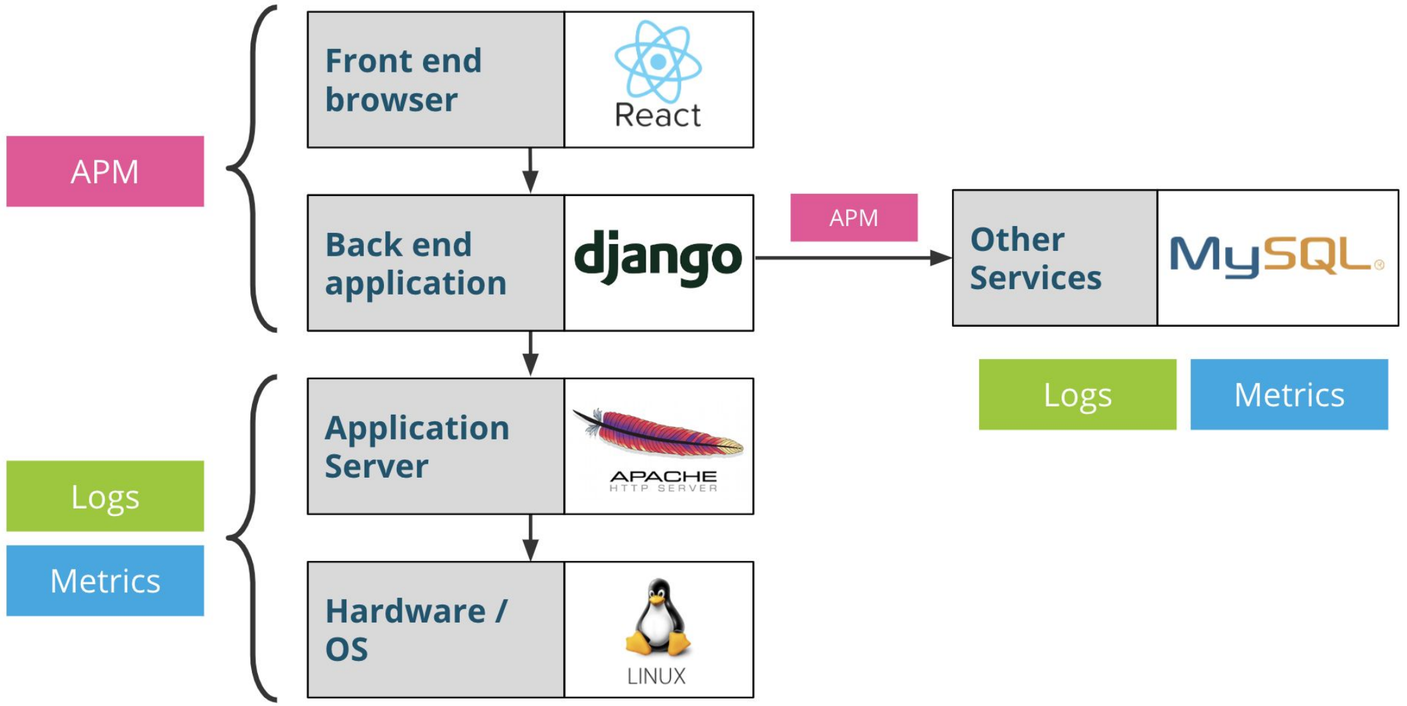


elastic

로그 + 메트릭 + 트레이스

통합 모니터링을 위해서는 세 가지 모두가 필요함

- 웹서버에 “로그”형태로 기록되는 사용자 요청과 에러 내용
- 운영체제 API 호출로 발생하는 CPU와 RAM 사용 “메트릭”
- 애플리케이션의 로딩 시간과 그 상세 내역이 기록된 “트레이스”

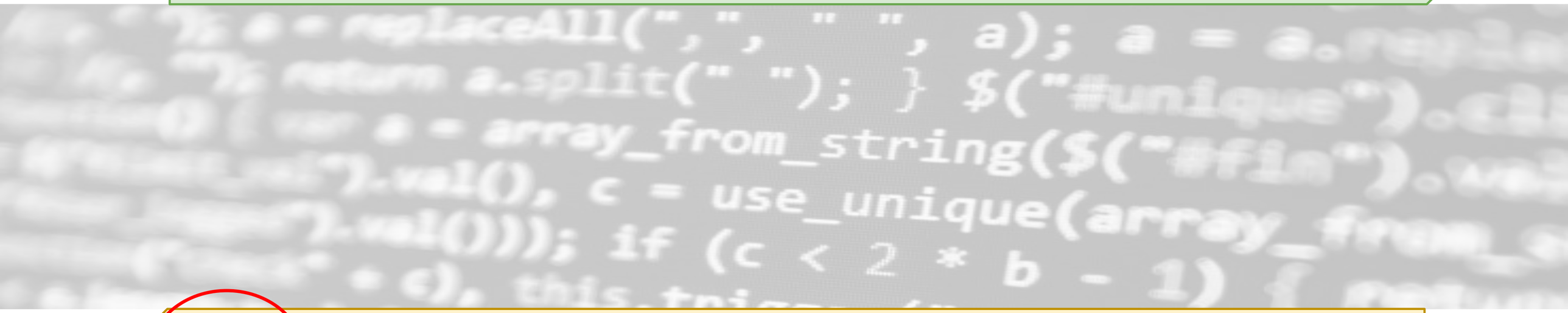


elastic

트레이스 데이터

예: 응답 시간 또는 실행 시간이 느린 경우

```
03:43:45 Request "GET cyclops.ESProductDetailView"
```



```
03:43:57 Response "cyclops.ESProductDetailView 200 OK"
```

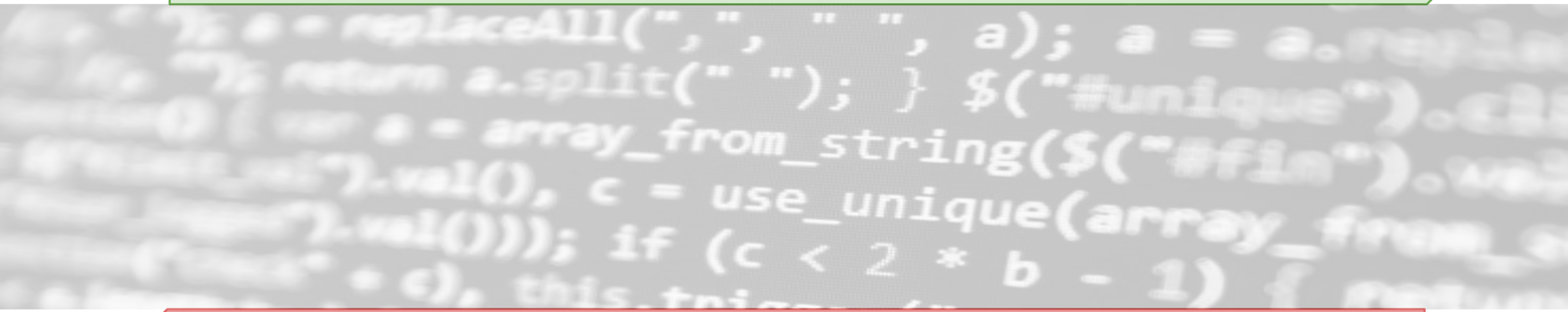
12 초

elastic

트레이스 데이터

예: 오류 및 예외 발생

03:43:59 Request "POST /api/checkout"



03:43:59 Response "/api/checkout 500 ERROR"

elastic

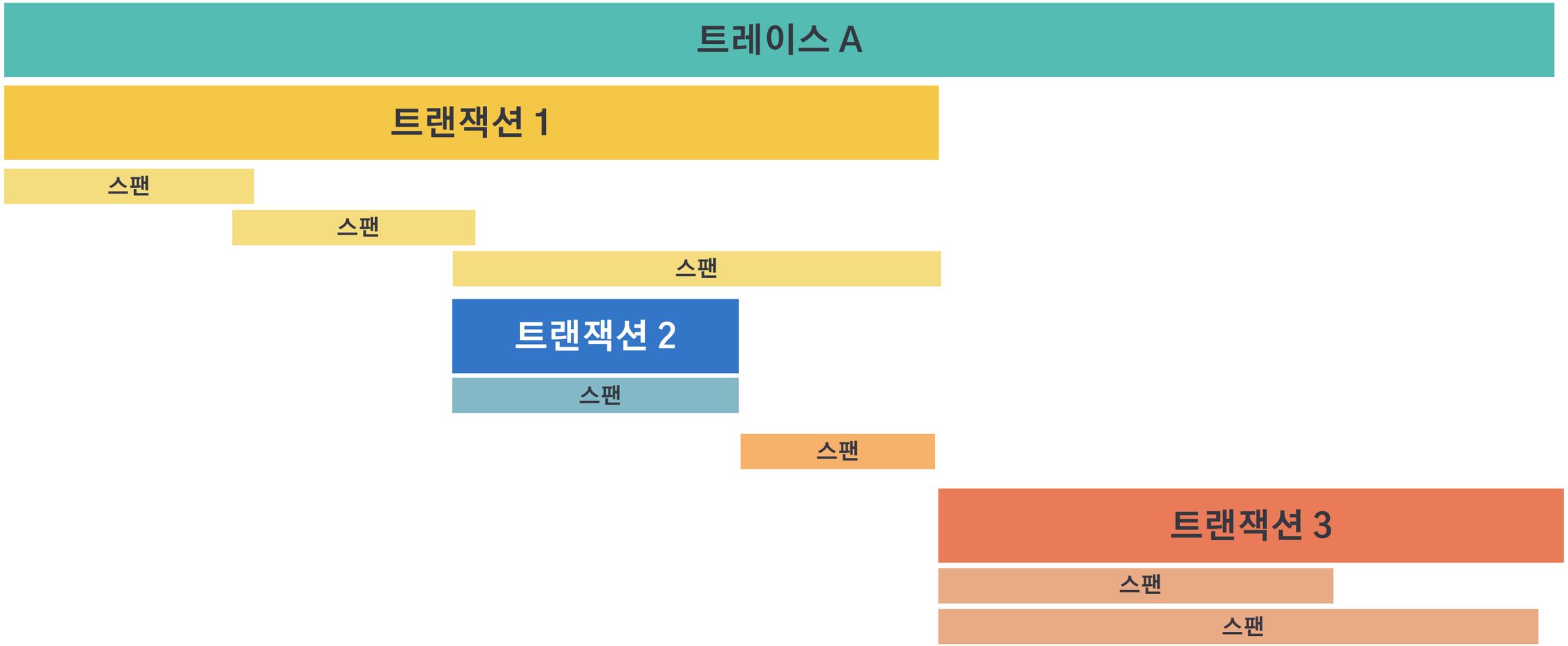
분산 트레이싱

단일 트랜잭션

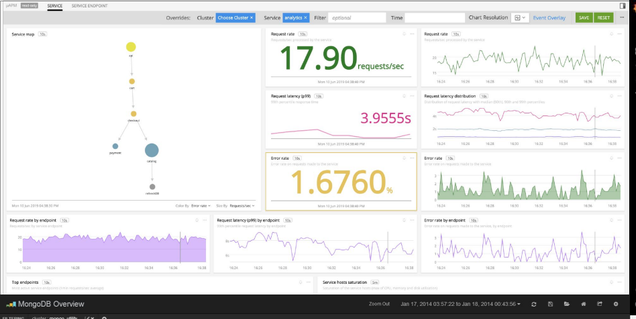
elastic

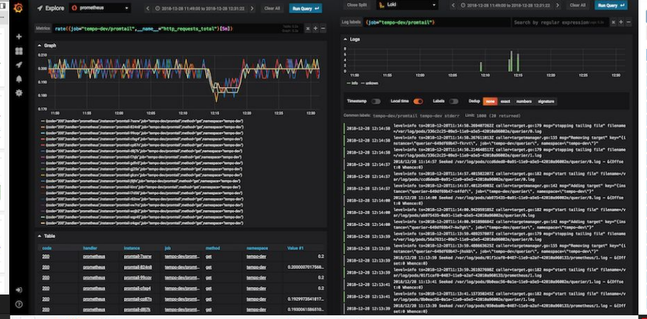
분산 트레이싱

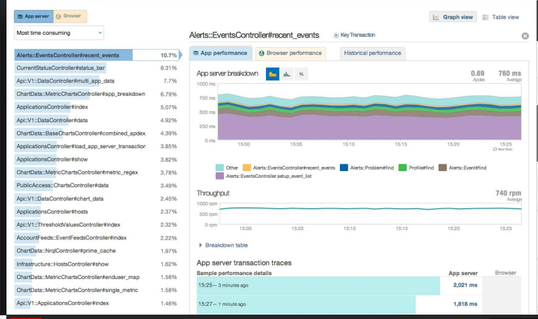
다수 서비스

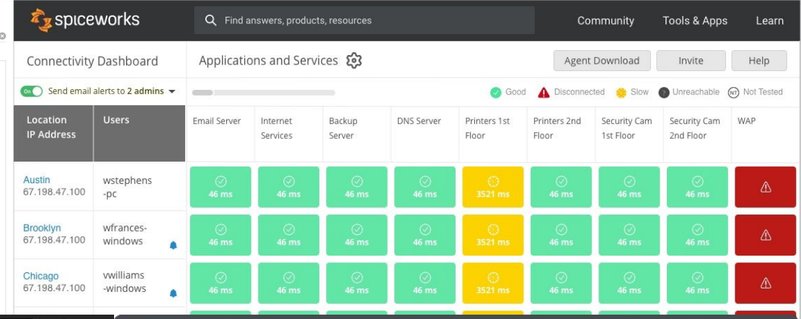


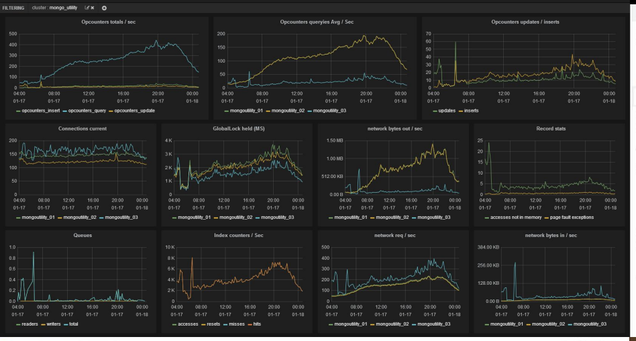
elastic

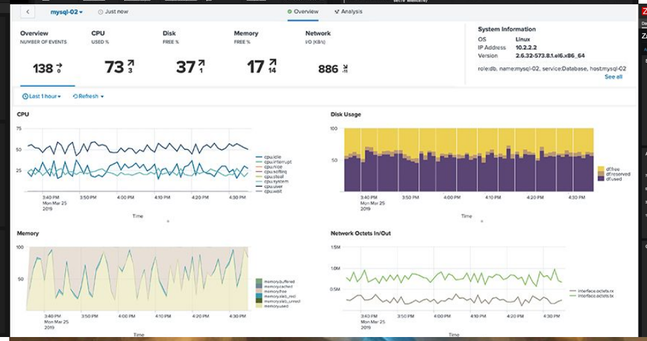

17.90 requests/sec
3.9555s
1.6760%

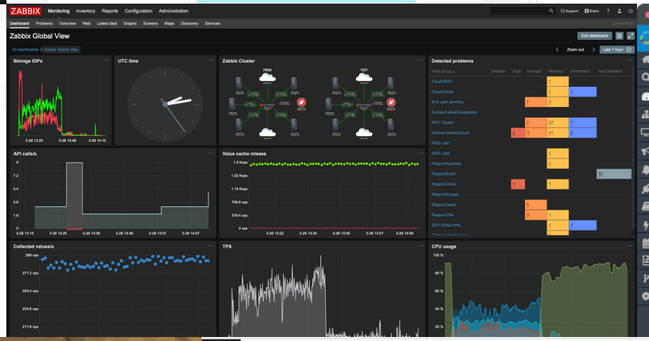


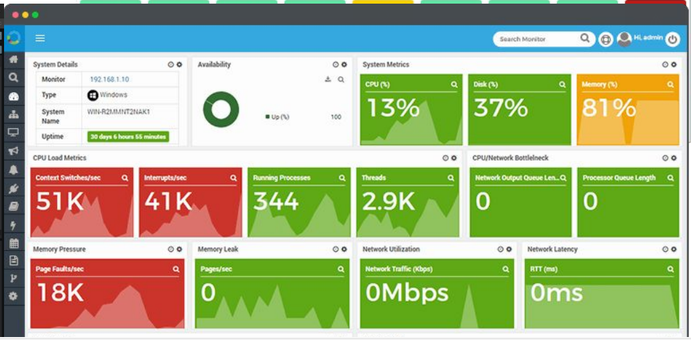



spiceworks
Find answers, products, resources
Community
Tools & Apps
Learn
Connectivity Dashboard
Applications and Services
Agent Download
Invite
Help
Location
IP Address
Users
Email Server
Internet Services
Backup Server
DNS Server
Printers 1st Floor
Printers 2nd Floor
Security Cam 1st Floor
Security Cam 2nd Floor
WAP
Austin
67.198.47.100
wstephens-pc
Brooklyn
67.198.47.100
wfrances-windows
Chicago
67.198.47.100
vwilliams-windows


MongoDB Overview

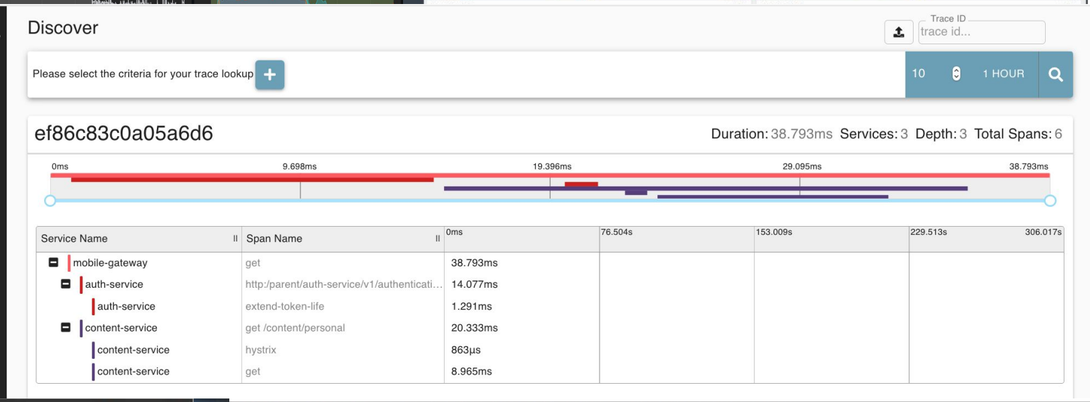

mysql-02
Overview
Analysis
System Information
CPU
Disk Usage
Memory
Network Octets In/Out

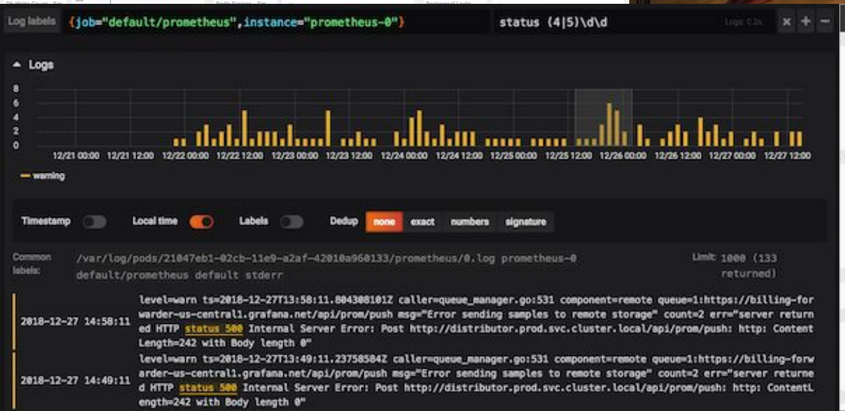

ZABBIX
Zabbix Global View

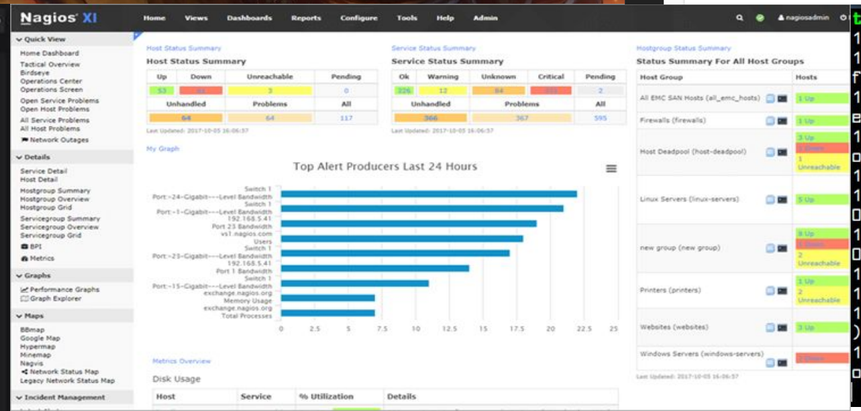

System Details
Availability
System Metrics
CPU (%)
13%
Disk (%)
37%
Memory (%)
81%
CPU Load Metrics
CPU/Network Bottleneck
Context Switches/sec
51K
Interrupts/sec
41K
Running Processes
344
Threads
2.9K
Network Output Queue Len...
0
Processor Queue Length
0
Memory Pressure
Memory Leak
Network Utilization
Network Latency
Page Faults/sec
18K
Pages/sec
0
Network Traffic (Kbps)
0Mbps
RTT (ms)
0ms

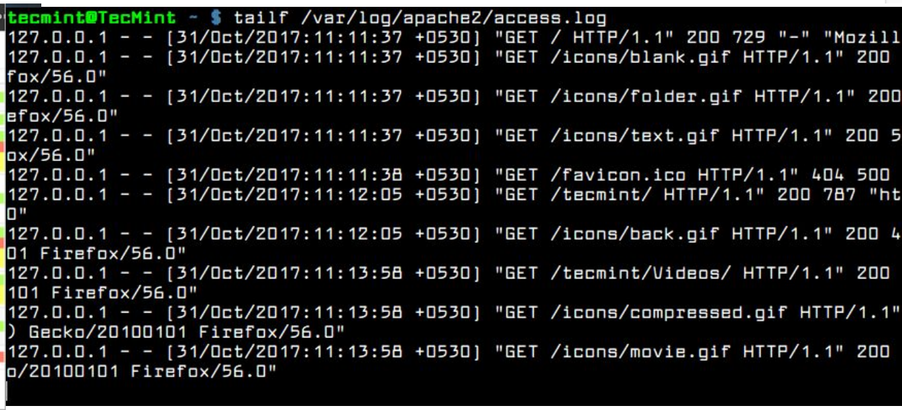

Screen Overview
1.22M
1.26M
1.26M
270.38
210.51
201.57
89.56/s
91.45/s
92.55/s
76.60M
2.15M
241.89

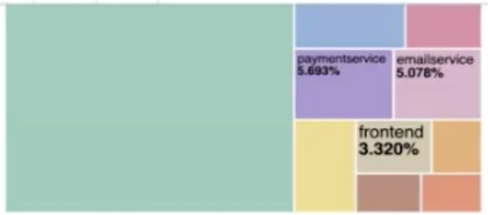

* 7015c90 - (HEAD, origin/master, origin/HEAD, master) Updated iPhones graphic with new screenshot (11 days ago) <DiNutter>
* 9bfa321 - Updated README (2 weeks ago) <DiNutter>
* dd18305 - Updated to new design (2 weeks ago) <DiNutter>
* b2e36c6 - Added fallback for missing avatar (2 weeks ago) <DiNutter>
* 8678978 - Removed Disk caching of avatar (6 weeks ago) <DiNutter>
* 39d2c3c - Preparing for App Store submission (9 weeks ago) <DiNutter>
* 4fd7261 - Fixed cell bg when user only has 1 badge. (9 weeks ago) <DiNutter>
* 13bf363 - Fix image (10 weeks ago) <DiNutter>
* 705363e - Try screenshots with transparent background (10 weeks ago) <DiNutter>
* bd4602e - Correct image location in README (10 weeks ago) <Will McKenzie>
* 483d05b - Added iPhone screenshots to README (10 weeks ago) <DiNutter>
* 2d7d6d7 - Adding license (10 weeks ago) <DiNutter>
* d6c6660 - Adding to .gitignore (10 weeks ago) <DiNutter>
* 0e2af71 - Still trying to ignore UserInterfaceState file (10 weeks ago) <DiNutter>
* cca3d3b - ignore interfaceState file (10 weeks ago) <DiNutter>
* 2b1e812 - Added first draft of README (10 weeks ago) <DiNutter>
* 38586ba - Bug fixes and UI Improvements. New Icons added (2 months ago) <DiNutter>
* 968d04c - Removed extra app icons (3 months ago) <DiNutter>
* ecf783d - Various updates (3 months ago) <DiNutter>
* 6ce53d5 - Removed SOURL files from root (3 months ago) <DiNutter>
* 62e0679 - Made sure images cache (3 months ago) <DiNutter>
* 1397df2 - Ran Arc conversion to improve memory management. Refactoring and bug fixes (3 months ago) <DiNutter>
* c3061b8 - Added pull to refresh functionality (3 months ago) <DiNutter>
* 926616b - Made sure iPhone badge and accomplishments view controllers reload when user changed. (3 months ago) <DiNutter>
* 83cd921 - Stopped badges images using separate threads. (3 months ago) <DiNutter>


Discover
Trace ID
trace id...
Please select the criteria for your trace lookup
10
1 HOUR
ef86c83c0a05a6d6
Duration: 38.793ms Services: 3 Depth: 6 Total Spans: 6
0ms
9.698ms
19.396ms
29.095ms
38.793ms
Service Name
Span Name
0ms
76.504s
153.009s
229.513s
306.017s
mobile-gateway get 38.793ms
auth-service http:/parent/auth-service/v1/authenticati... 14.077ms
auth-service extend-token-life 1.291ms
content-service get /content/personal 20.333ms
content-service hystrix 863µs
content-service get 8.965ms


Log labels
{job="default/prometheus",instance="prometheus-0"}
status (4|5)\d\d
Logs
warning
Timestamp
Local time
Labels
Dedup
none
exact
numbers
signature
Common labels:
/var/log/pods/21047eb1-02cb-11e9-a2af-42010a960133/prometheus/0.log prometheus-0 default/prometheus default stderr
Limit: 1000 (133 returned)
2018-12-27 14:58:11
level=warn ts=2018-12-27T13:58:11.804300101Z caller=queue_manager.go:531 component=remote queue=1:https://billing-forwarder-us-central1.grafana.net/api/prom/push msg="Error sending samples to remote storage" count=2 err="server returned HTTP status 500 Internal Server Error: Post http://distributor.prod.svc.cluster.local/api/prom/push: http: ContentLength=242 with Body length 0"
2018-12-27 14:49:11
level=warn ts=2018-12-27T13:49:11.237585842Z caller=queue_manager.go:531 component=remote queue=1:https://billing-forwarder-us-central1.grafana.net/api/prom/push msg="Error sending samples to remote storage" count=2 err="server returned HTTP status 500 Internal Server Error: Post http://distributor.prod.svc.cluster.local/api/prom/push: http: ContentLength=242 with Body length 0"


Nagios XI
Home
Views
Dashboards
Reports
Configure
Tools
Help
Admin
Host Status Summary
Service Status Summary
Hostgroup Status Summary
Status Summary For All Host Groups
Top Alert Producers Last 24 Hours
Disk Usage


tecmint@TecMint ~ $ tailf /var/log/apache2/access.log
127.0.0.1 - - [31/Oct/2017:11:11:37 +0530] "GET / HTTP/1.1" 200 729 "-" "Mozill
127.0.0.1 - - [31/Oct/2017:11:11:37 +0530] "GET /icons/blank.gif HTTP/1.1" 200
fox/56.0"
127.0.0.1 - - [31/Oct/2017:11:11:37 +0530] "GET /icons/folder.gif HTTP/1.1" 200
efox/56.0"
127.0.0.1 - - [31/Oct/2017:11:11:37 +0530] "GET /icons/text.gif HTTP/1.1" 200 5
ox/56.0"
127.0.0.1 - - [31/Oct/2017:11:11:38 +0530] "GET /favicon.ico HTTP/1.1" 404 500
127.0.0.1 - - [31/Oct/2017:11:12:05 +0530] "GET /tecmint/ HTTP/1.1" 200 787 "ht
0"
127.0.0.1 - - [31/Oct/2017:11:12:05 +0530] "GET /icons/back.gif HTTP/1.1" 200 4
01 Firefox/56.0"
127.0.0.1 - - [31/Oct/2017:11:13:58 +0530] "GET /tecmint/Videos/ HTTP/1.1" 200
101 Firefox/56.0"
127.0.0.1 - - [31/Oct/2017:11:13:58 +0530] "GET /icons/compressed.gif HTTP/1.1
) Gecko/20100101 Firefox/56.0"
127.0.0.1 - - [31/Oct/2017:11:13:58 +0530] "GET /icons/movie.gif HTTP/1.1" 200
o/20100101 Firefox/56.0"



Percentile
0.67 ms
Percentile (50)

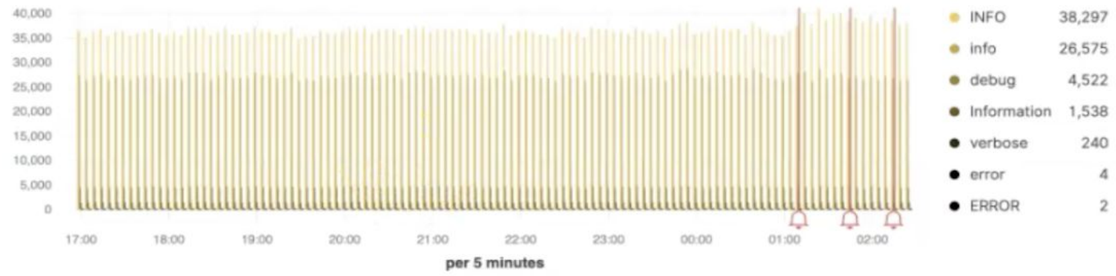
Percentile
107.43 ms
Percentile (90)

364,050
Count

5,533
Products Shipped

Logs

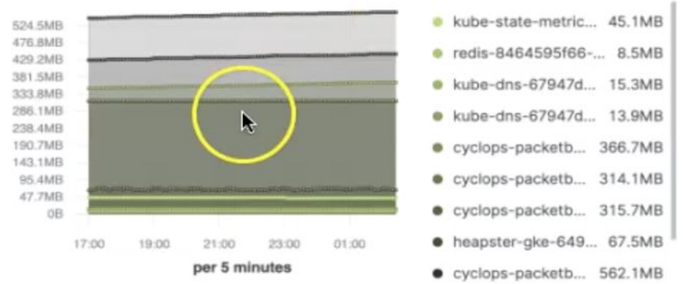
Log Levels & Outages



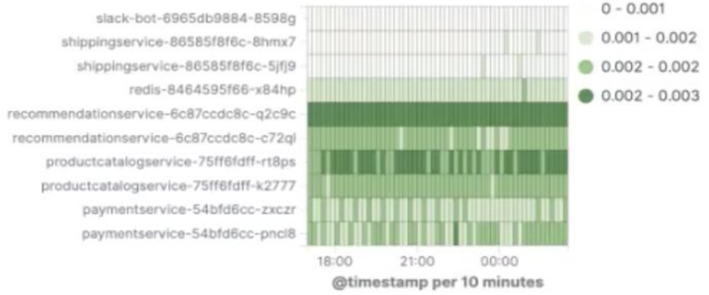
Logs by Service

Metrics

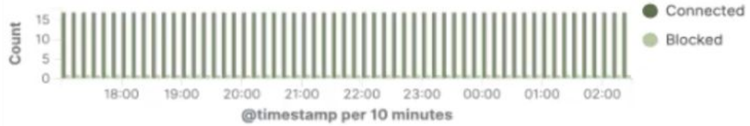
Memory Usage



CPU Usage



Connected clients [Metricbeat Redis] ECS

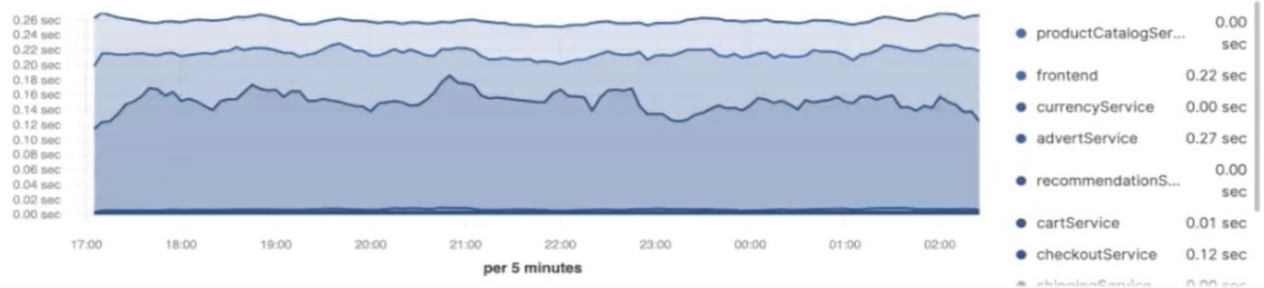


Hosts [Metricbeat Redis] ECS

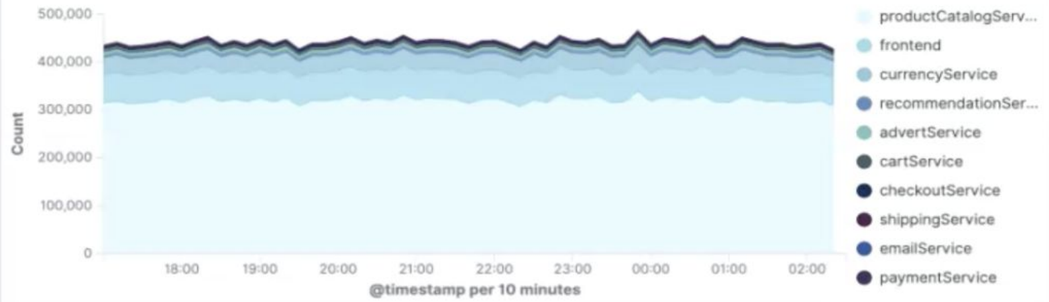
service.address: Descending	Uptime (s)	PID	Memory	CPU used (user)	CPU used (system)
10.80.9.59:6379	42,046	1	1,448,096	61.744	72.956

APM

APM Service Performance



APM Number of Requests



CPU, Memory and SLA

Elastic Solutions

Elastic Enterprise Search
엘라스틱 엔터프라이즈 검색

Elastic Observability
엘라스틱 통합 가시성

Elastic Security
엘라스틱 보안

Elastic Stack
엘라스틱 스택

elastic

Logs + Metrics + APM + Uptime + User Exp’

Elastic Observability

통합 가시성

단일 플랫폼으로
문제 해결 시간(MTTR)을 최소화, 비용도 최소화

elastic

Demo

One-stop for Hipster Fashion & Style Online

https://github.com/GoogleCloudPlatform/microservices-demo

Online BOUTIQUE EST. 2015

Continuous Integration - Master/Release passing

Online Boutique is a cloud-native microservices demo application. Online Boutique consists of a 10-tier microservices application. The application is a web-based e-commerce app where users can browse items, add them to the cart, and purchase them.

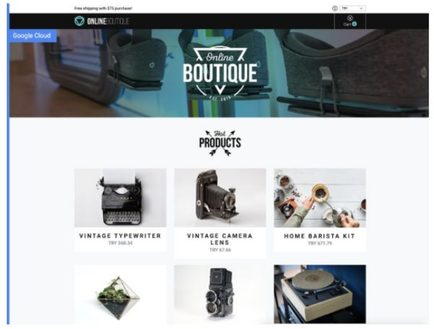
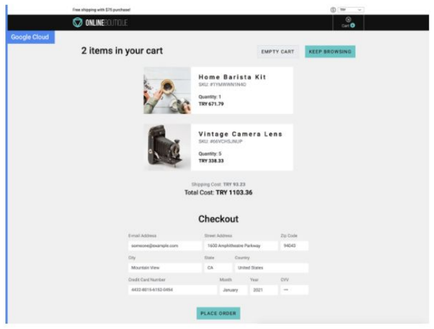
Google uses this application to demonstrate use of technologies like Kubernetes/GKE, Istio, Stackdriver, gRPC and OpenCensus. This application works on any Kubernetes cluster, as well as Google Kubernetes Engine. It's **easy to deploy with little to no configuration**.

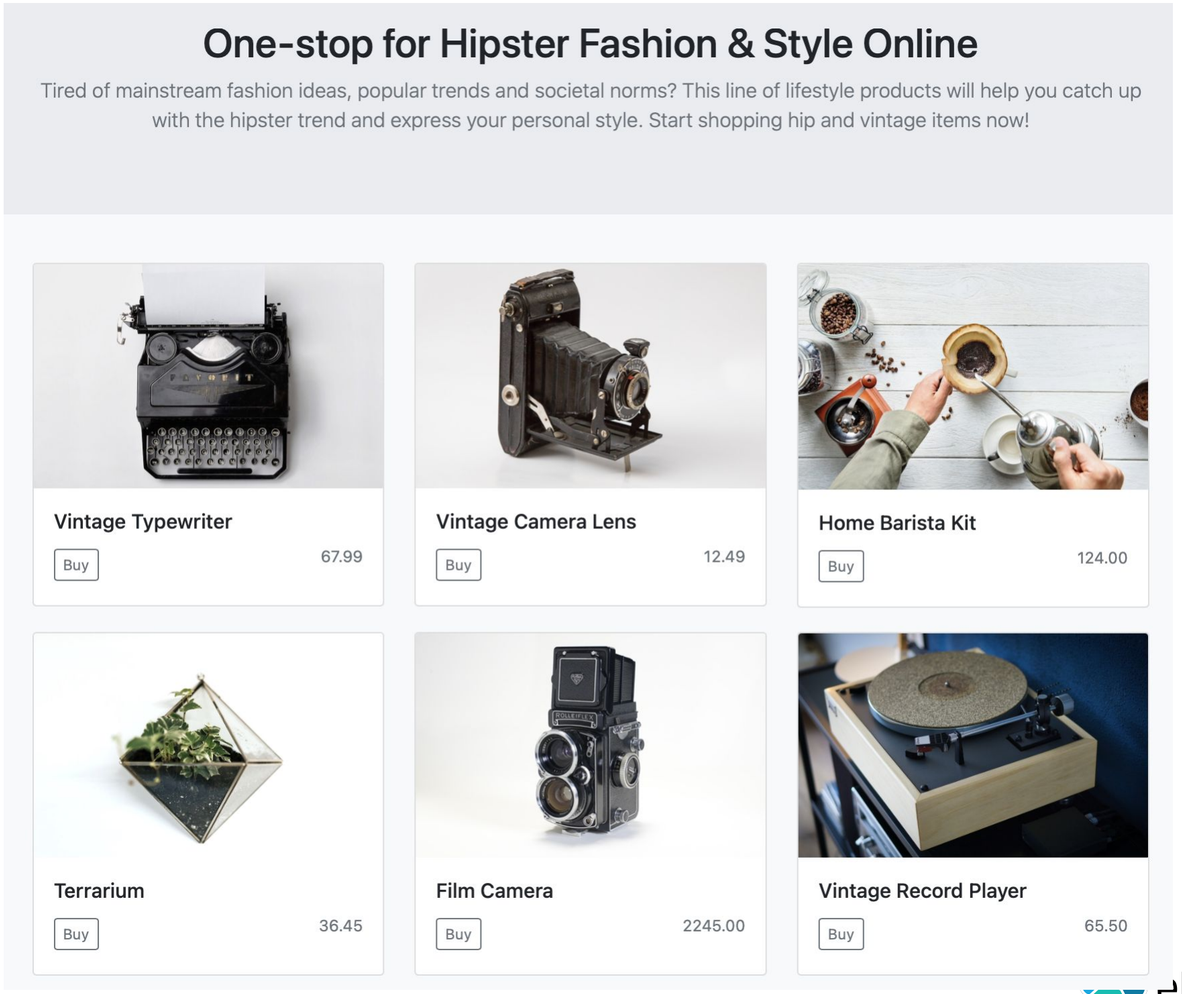
If you're using this demo, please ★**Star** this repository to show your interest!

> 👓**Note to Googlers:** Please fill out the form at go/microservices-demo if you are using this application.

Looking for the old Hipster Shop frontend interface? Use the manifests in release v0.1.5.

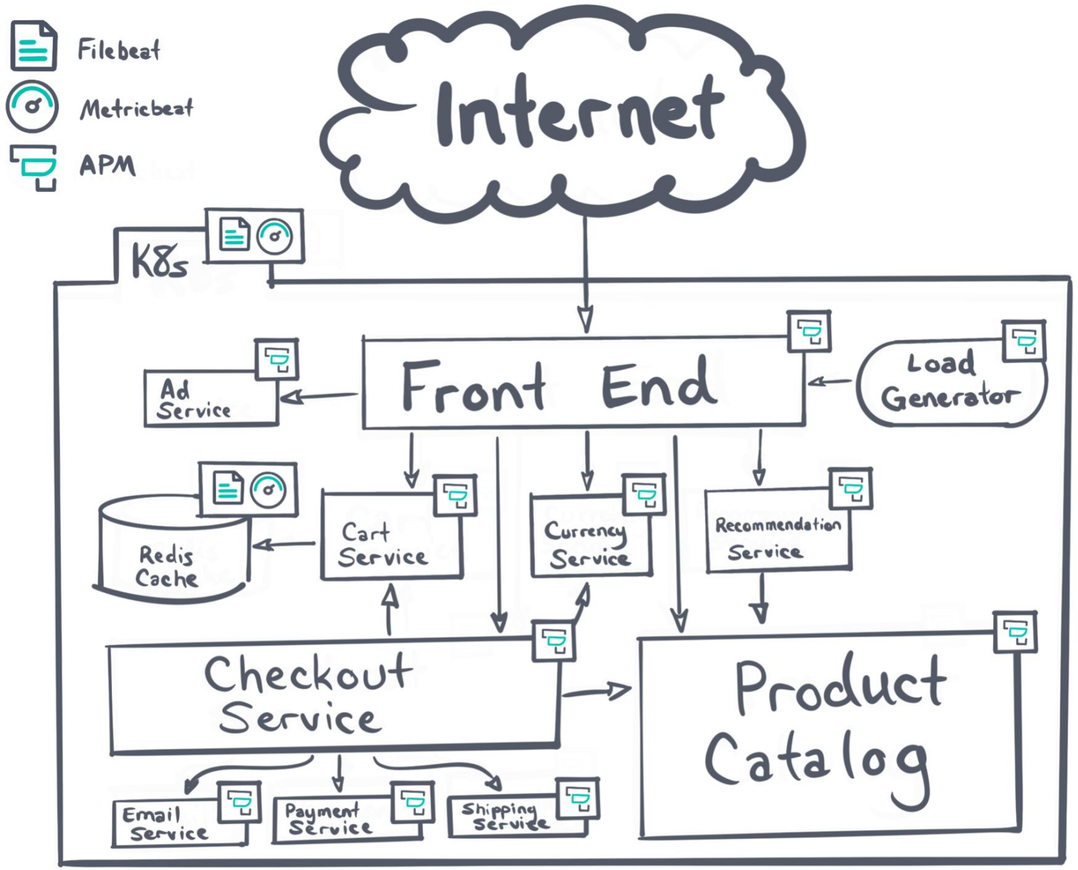
Screenshots

Home Page	Checkout Screen
	



elastic

Architecture



Service	Language	Description
frontend	Go	Exposes an HTTP server to serve the website. Does not require signup/login and generates session IDs for all users automatically.
cartservice	C#	Stores the items in the user's shopping cart in Redis and retrieves it.
productcatalogservice	Go	Provides the list of products from a JSON file and ability to search products and get individual products.
currencyservice	Node.js	Converts one money amount to another currency. Uses real values fetched from European Central Bank. It's the highest QPS service.
paymentservice	Node.js	Charges the given credit card info (mock) with the given amount and returns a transaction ID.
shippingservice	Go	Gives shipping cost estimates based on the shopping cart. Ships items to the given address (mock)
emailservice	Python	Sends users an order confirmation email (mock).
checkoutservice	Go	Retrieves user cart, prepares order and orchestrates the payment, shipping and the email notification.
recommendationservice	Python	Recommends other products based on what's given in the cart.
adservice	Java	Provides text ads based on given context words.
loadgenerator	Python/Locust	Continuously sends requests imitating realistic user shopping flows to the frontend.

elastic

감사합니다!

Q&A