

# Pinpoint 로 어플리케이션에 대한 Naver 수준의 가시성 확보하기

김성욱  
정현길

# Index

---

- Microservice Architecture
- Observability in Naver
- Enhance Observability with Pinpoint
- Open source Pinpoint
- Open source Community
- Troubleshooting Distributed Systems
- How Pinpoint Works
- Troubleshooting with Pinpoint
- Future Plans

# **Microservice Architecture**

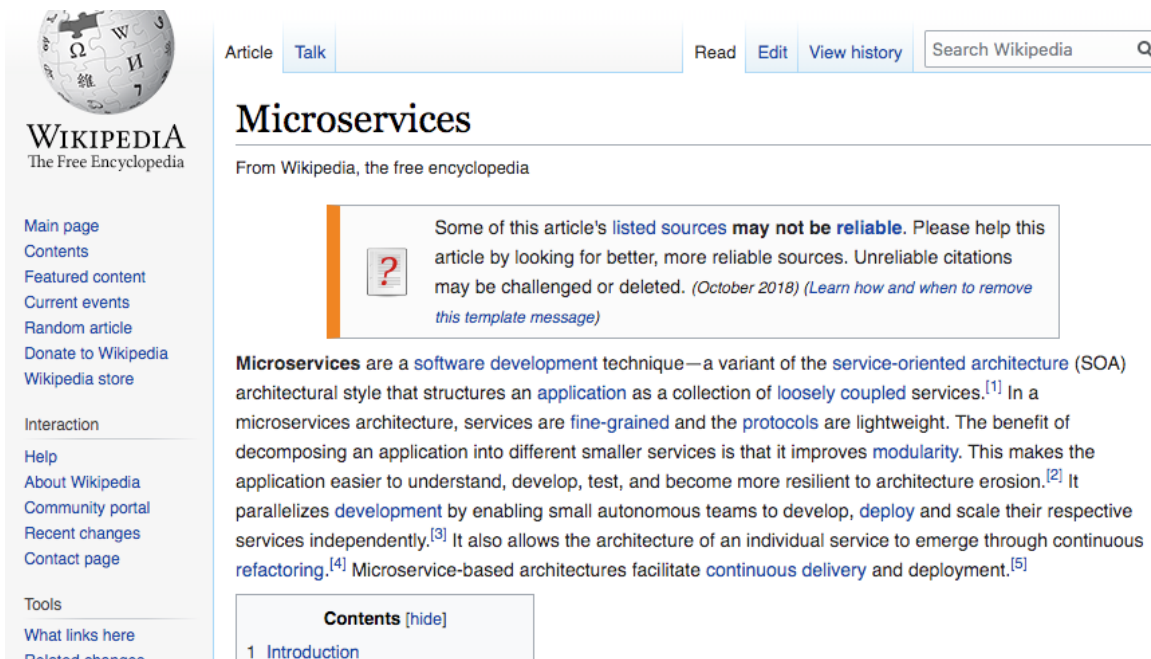
# Microservice Architecture

---

- Development
- Maintenance
- Reliability
- Scalability
- Cost
- Deployment
- Releasing

# Microservice Architecture

- Development
- Maintenance
- Reliability
- Scalability
- Cost
- Deployment
- Releasing



The screenshot shows the Wikipedia page for 'Microservices'. At the top left is the Wikipedia logo and the text 'WIKIPEDIA The Free Encyclopedia'. Below this is a sidebar with navigation links: Main page, Contents, Featured content, Current events, Random article, Donate to Wikipedia, and Wikipedia store. Under 'Interaction' are links for Help, About Wikipedia, Community portal, Recent changes, and Contact page. Under 'Tools' are links for What links here and Related changes. The main content area has tabs for 'Article' and 'Talk', and buttons for 'Read', 'Edit', and 'View history'. A search bar is located at the top right. The title 'Microservices' is prominently displayed. Below the title is the text 'From Wikipedia, the free encyclopedia'. A warning box with a question mark icon states: 'Some of this article's listed sources may not be reliable. Please help this article by looking for better, more reliable sources. Unreliable citations may be challenged or deleted. (October 2018) (Learn how and when to remove this template message)'. The main text begins with: 'Microservices are a software development technique—a variant of the service-oriented architecture (SOA) architectural style that structures an application as a collection of loosely coupled services.<sup>[1]</sup> In a microservices architecture, services are fine-grained and the protocols are lightweight. The benefit of decomposing an application into different smaller services is that it improves modularity. This makes the application easier to understand, develop, test, and become more resilient to architecture erosion.<sup>[2]</sup> It parallelizes development by enabling small autonomous teams to develop, deploy and scale their respective services independently.<sup>[3]</sup> It also allows the architecture of an individual service to emerge through continuous refactoring.<sup>[4]</sup> Microservice-based architectures facilitate continuous delivery and deployment.<sup>[5]</sup>' Below the text is a 'Contents' section with a 'hide' link and a list item '1 Introduction'.

# Microservice Architecture

- **Development**
- Maintenance
- Reliability
- Scalability
- Cost
- **Deployment**
- Releasing

The image shows a screenshot of the Wikipedia article for 'Microservices'. The article title is 'Microservices' and it is described as 'From Wikipedia, the free encyclopedia'. A red box highlights a sentence in the text: 'Microservice-based architectures facilitate continuous delivery and deployment.' This sentence is also underlined in red. A warning box is visible above the highlighted text, stating: 'Some of this article's listed sources may not be reliable. Please help this article by looking for better, more reliable sources. Unreliable citations may be challenged or deleted. (October 2018) (Learn how and when to remove this template message)'. The Wikipedia interface includes a search bar, navigation tabs (Article, Talk), and a sidebar with links like 'Main page', 'Contents', and 'Tools'.

**Microservice-based architectures facilitate continuous delivery and deployment.**

## New Research Shows 63% of Enterprises Are Adopting Microservices Architectures

Despite new research, 50% of enterprises are unaware of microservices' impact on revenue-generating business processes.



by Tom Smith · Sep. 20, 18 · Microservices Zone · Interview

 Like (4)

 Comment (0)

 Save

 Tweet

 4,775 Views

Join the DZone community and get the full member experience.

[JOIN FOR FREE](#)

Download DZone's 2019 Scaling DevOps Trend Report to learn how to ensure security as you scale DevOps, why "blameless retrospectives" are a myth, and the key considerations for getting the most out of DevOps as you scale.

[Download Now](#)

Presented by DZone

---

<https://dzone.com/articles/right-strategies-for-microservices-deployment>

# Microservice Architecture

coupage TECH NOTES EVENTS

행복을 찾기 위한 우리의 여정,

Coupage technology blog Team  
May 25, 2018 - 12 min read

쿠팡의 MSA — Part 1

“행복을 찾기 위한 우리의 여정, 쿠팡의 MSA”에 대하여 2가지이다.

- Micro service architecture를 위한 쿠팡의 여정

BLOG TECH Tony Mauro of NGINX, Inc. February 19, 2015

## Adopting Microservices at Netflix: Lessons for Architectural Design

microservices, containers, Netflix, content delivery network (CDN)

TWITTER LINKEDIN Y F

쿠팡의 MSA를 기술 블로그

그 결과로 지금은 Confluence를 이용한 문서화 수준을 넘어서, 각 프로젝트마다 특성에 맞게 Scrum 또는 Kanban Board를 만들고, 업무들의 진행 상황을 같이 살펴 보고 논의하며 일을 하고 있습니다.

### Java로의 전환

가끔 회사 외부 분들을 만나서 대화를 나누다 보면, 배달의민족은 어떤 언어를 이용하여 구현했나는 질문을 받는데요. 크게 보면 대부분의 서비스 로직이 MS SQL Server라는 DBMS의 Stored Procedure로 구성되어 있고, API 요청을 받는 쪽은 PHP로 구현이 되어 있습니다.

DBMS에 모든 부하가 몰리는 구조이기 때문에 서비스 요청이 많아지면서 특정 요청에 대한 처리가 잘못되면 DBMS 부하로 장애가 나타나지는 현상이 많았습니다. 근본적인 해결책은 DBMS에 있는 서비스 로직을 레밍 언어를 쓸까 고민하다가 Java를 선택하게 되었습니다. (왜 Java를 선택했는지, DBMS와 밀접하게 연결되어 있는 빌링 모듈을 별도로 개발한 이유 등) 이미 PHP로 개발이 진행된 부분이 있었음에도 불구하고, 해당 모듈을 Java로 개발하는 것은 상당한 리스크를 수반하는 결정이었습니다. 하지만, 이 결정은 결국에는 옳은 결정이었습니다. 그 뒤로는... 프로젝트 진행 도중에 일어난 잘못된 의사 결정이 얼마 없었습니다. 그리고 동시에 배달의민족 서비스를 Micro-service architecture 형태로 전환했습니다.

In our recent blog posts, we've explained why we believe it's crucial to adopt a four-tier application architecture in which applications are developed and deployed as sets of microservices. It's becoming increasingly clear that if you keep using development processes and application architectures that worked just a few years ago, you simply can't move fast enough to capture and hold the interest of mobile users who come from an ever-growing number of apps.

Adopting a microservices architecture creates exciting opportunities in the marketplace for companies. For architects and developers, it promises an unprecedented level of control and speed as they deliver innovative new web experiences to customers. But at such a breathless pace, it can feel like there's not a lot of room for error. In the real world, you can't stop developing and deploying your apps as you retool the processes

LG CNS

All CNS Story IT Insight IT Solutions IT Life

IT Insight

블록을 조합하듯 앱을 조합하는 '마이크로서비스' ②

2016.12.12 09:30

[연재 기획] 최신 IT를 만나다 (4-2편)

LG CNS의 시나 전문가들이 LG CNS 블로그 독자 여러분들을 위해 최신 IT 기술 및 트렌드를 소개해 드립니다.

매월 1회씩 아래와 같은 순서로 연재될 예정입니다. 독자 여러분의 많은 관심과 기대 바랍니다.

[연재 기획 주제]

- 1편: IT를 통해 서비스업으로 변화하고 있는 제조업체들
- 2편: 무엇을 해야 할지도 알려주는 처방형식의 세계
- 3편: 열의 시대에서 가상 비서의 시대로, Virtual Personal Assistant
- 4-1편: 블록을 조합하듯 앱을 조합하는 '마이크로서비스' ①
- 4-2편: 블록을 조합하듯 앱을 조합하는 '마이크로서비스' ②

[연재 주제는 기고 시점의 이슈, IT 트렌드에 따라 바뀔 수 있습니다]

지난 시간에는 마이크로서비스의 정의와 특징 및 주목받게 된 배경과 마이크로서비스 도입의 주요 목적에 대해 살펴보았습니다. 이번 시간에는 마이크로서비스의 구성 요소, 구조와 구현 방법, 그리고 도입 시 고려해야 할 사항에 대해 알아보겠습니다.

Samsung Connect 마이크로 서비스 사례

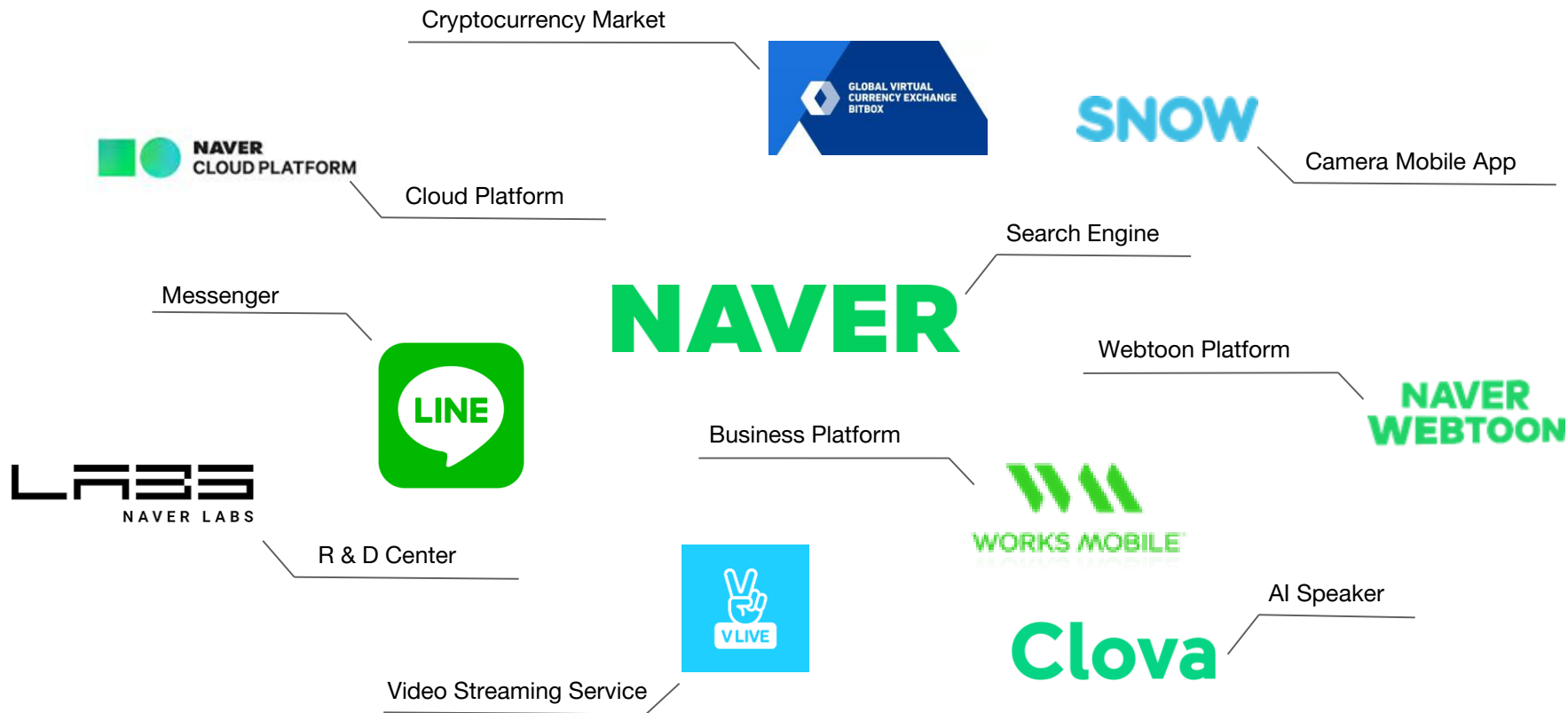
<b>Challenges</b>	<b>Solutions</b>
Massive traffic	EC2 Containers & Auto Scaling
4 different global teams	Autonomous/Decentralized
60+ application modules	Monolithic to Fine-grained
Secure, scalable and reliable	Automation and Independent deployment

**Microservices Architecture on AWS**

(c) 장수백(삼성전자), Samsung Connect 마이크로서비스 도입 사례, AWS Summit 2016



# Microservice Architecture



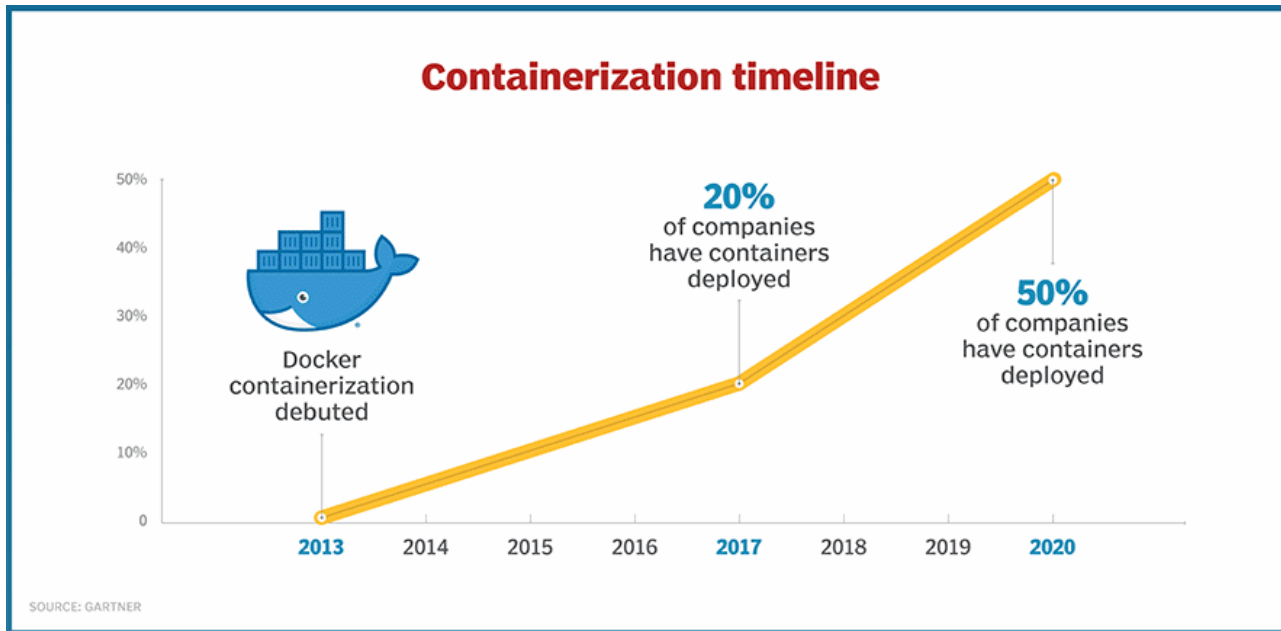
# Microservice Architecture

---

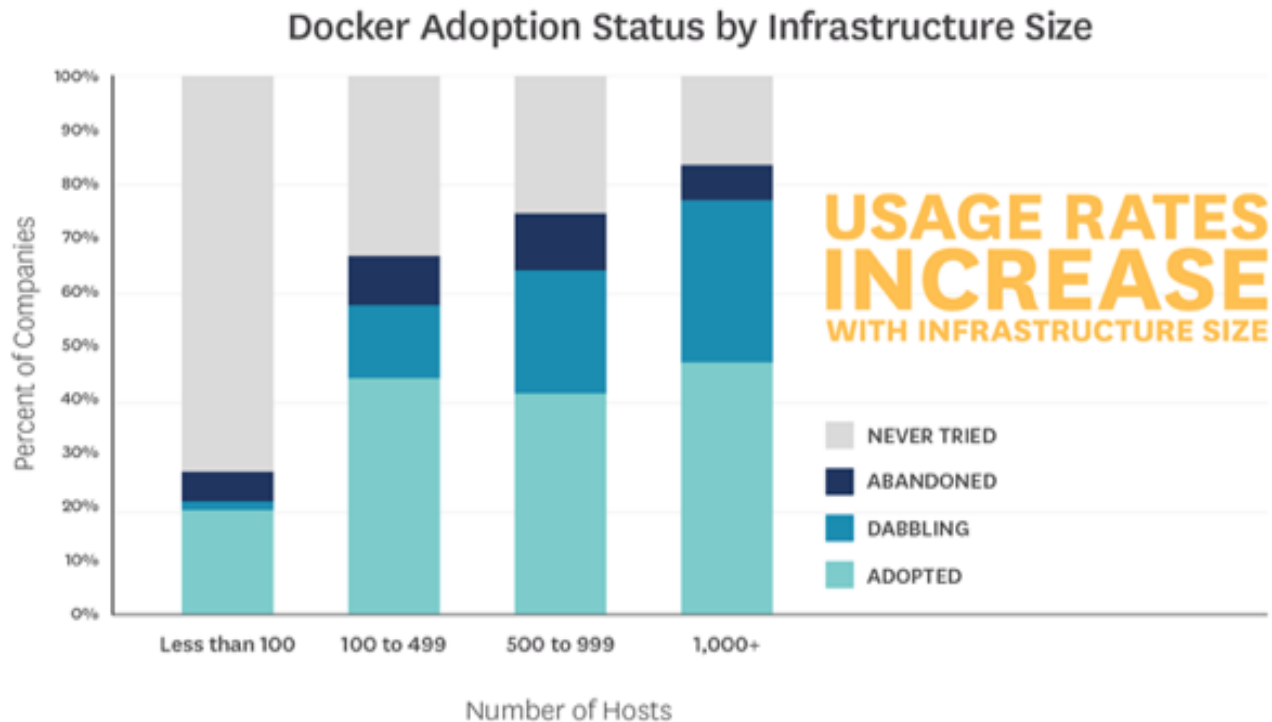
- Deployment Management
- Debugging

# Microservice Architecture

- ~~Deployment Management~~
- Debugging

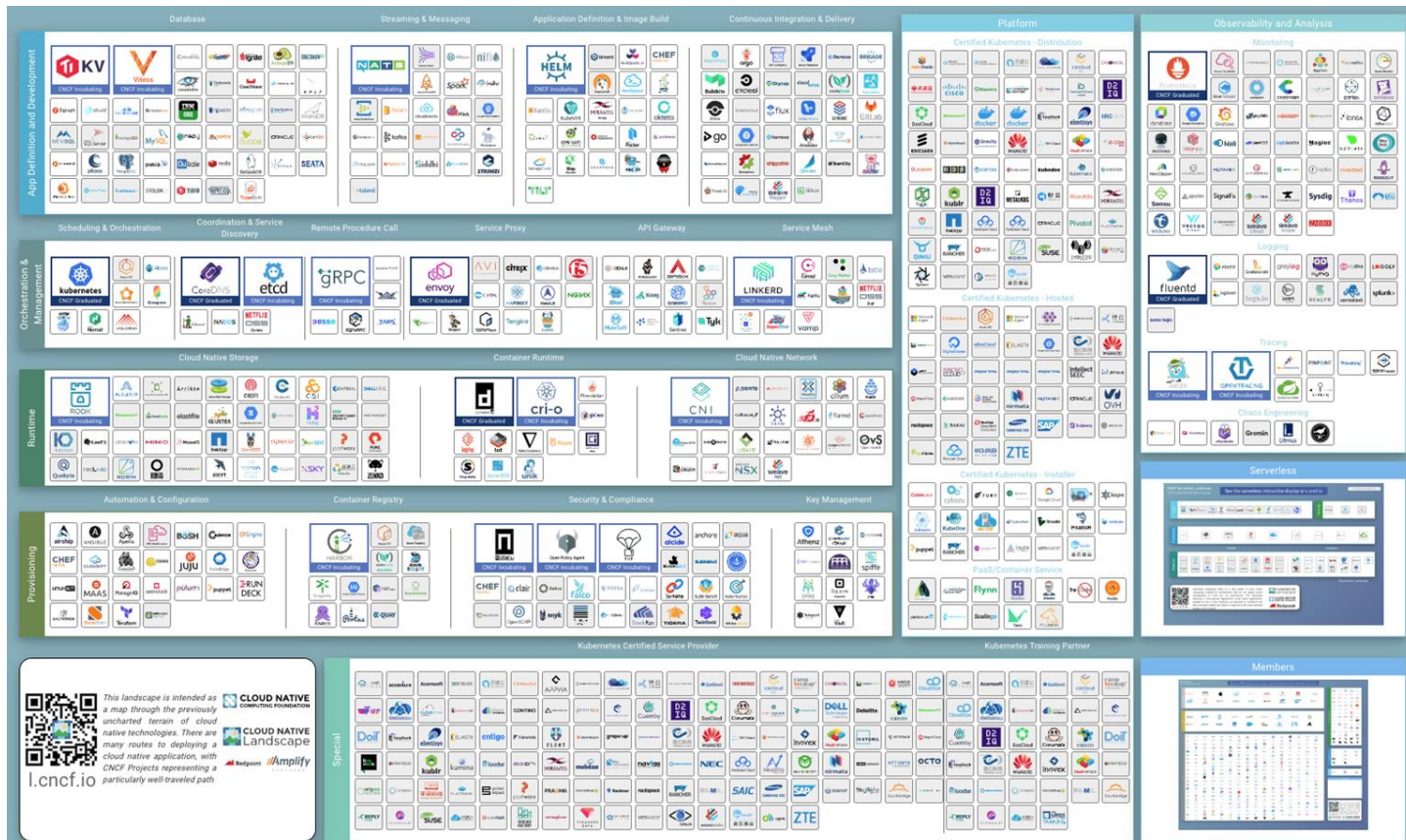


# Microservice Architecture



Source: Datadog

# Microservice Architecture



l.cncf.io

This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.

**CLOUD NATIVE**  
COOPERATIVE FOUNDATION

**CLOUD NATIVE**  
LANDSCAPE

Amplify

Special

# Microservice Architecture

The image displays a vast collection of logos for various microservice architecture components, organized into several categories:

- App Definition and Development:** Includes logos for KV, V, and various CI/CD tools.
- Database:** Features logos for KV, V, and other database providers.
- Streaming & Messaging:** Includes logos for NAT, HELM, and other messaging services.
- Application Definition & Image Build:** Features logos for HELM, CHEF, and other build tools.
- Continuous Integration & Delivery:** Includes logos for Jenkins, GitLab, and other CI/CD tools.
- Platform:** Divided into Certified Kubernetes - Distribution and Certified Kubernetes - Hosted, featuring logos for various Kubernetes distributions and hosts.
- Observability and Analysis:** Includes logos for Monitoring, Logging, and other observability tools. A red box highlights a specific logo in this section.
- Orchestration & Management:** Features logos for Kubernetes, CoreDNS, etcd, and other orchestration tools.
- Scheduling & Orchestration:** Includes logos for Kubernetes, CoreDNS, etcd, and other scheduling tools.
- Coordination & Service Discovery:** Features logos for gRPC, envoy, and other coordination services.
- Remote Procedure Call:** Includes logos for gRPC, envoy, and other RPC services.
- Service Proxy:** Features logos for envoy, nginx, and other service proxies.
- API Gateway:** Includes logos for various API gateway providers.
- Service Mesh:** Features logos for Linkerd, Istio, and other service mesh providers.
- Cloud Native Storage:** Includes logos for various cloud storage providers.
- Container Runtime:** Features logos for Docker, cri-o, and other container runtimes.
- Automation & Configuration:** Includes logos for various automation and configuration tools.
- Container Registry:** Features logos for various container registry providers.
- Security & Compliance:** Includes logos for various security and compliance tools.
- Key Management:** Features logos for various key management providers.
- Provisioning:** Includes logos for various provisioning tools.
- Members:** A section at the bottom right listing various member organizations.

A large **PINPOINT** logo is overlaid in the center of the grid. A red box highlights a specific logo in the Observability and Analysis section, which is identified as **PINPOINT** in the original image.

**Special:** A QR code and text are located in the bottom left corner, mentioning **l.cncf.io** and **CLOUD NATIVE COOPERATIVE FOUNDATION**.

# Microservice Architecture

---

- ~~Deployment Management~~
- Debugging

# Observability in Naver

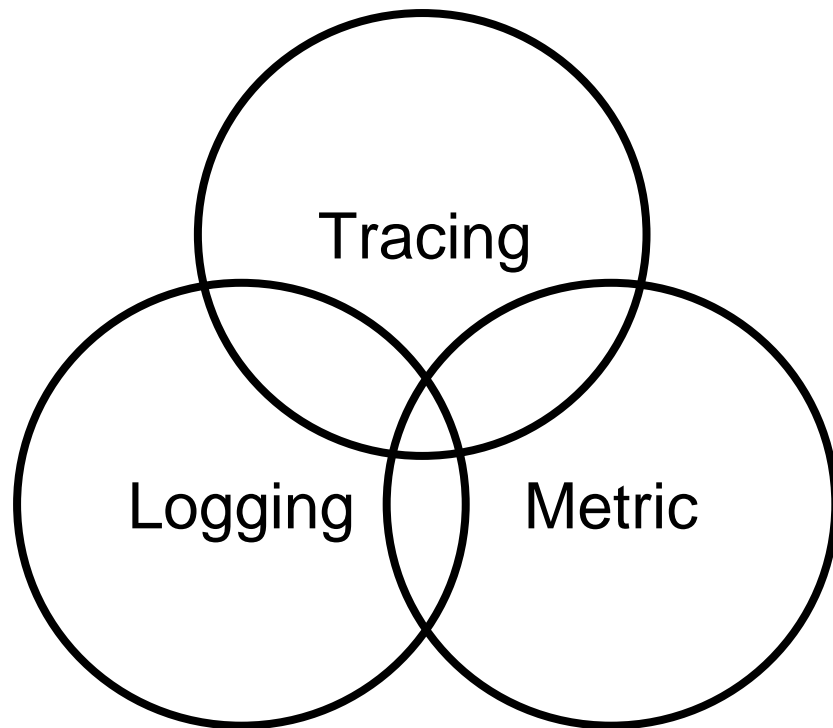
---

- ~~Deployment Management~~
- ~~Debugging~~

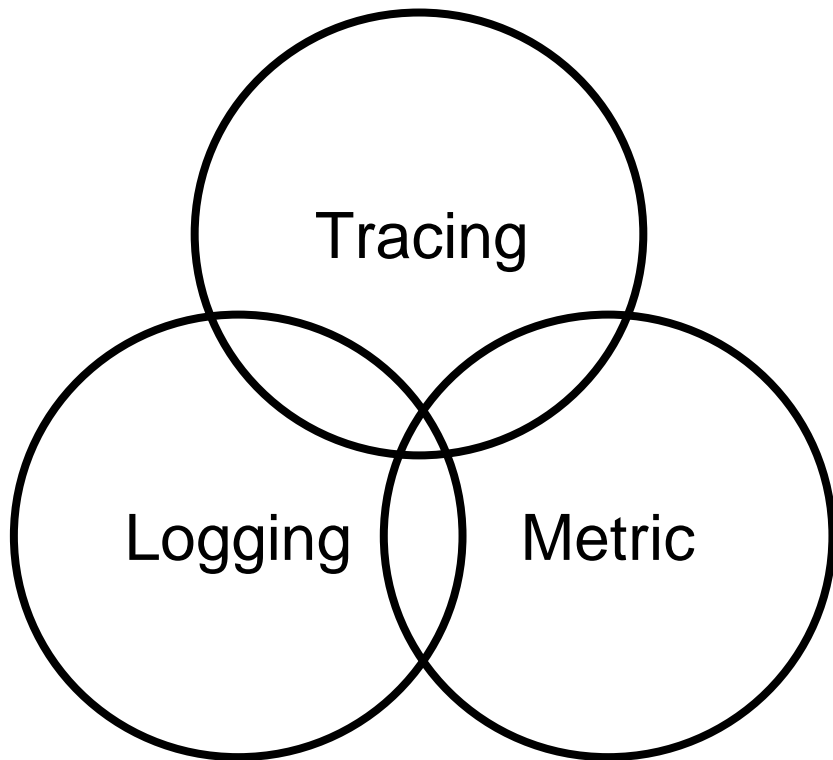
**Observability** ↑



## Observability

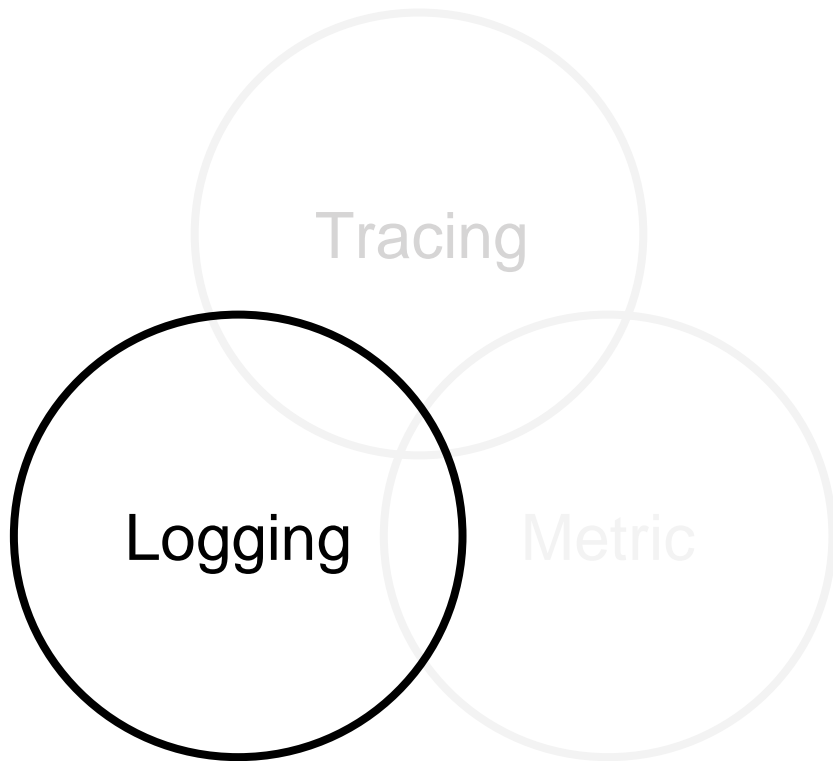


## Observability



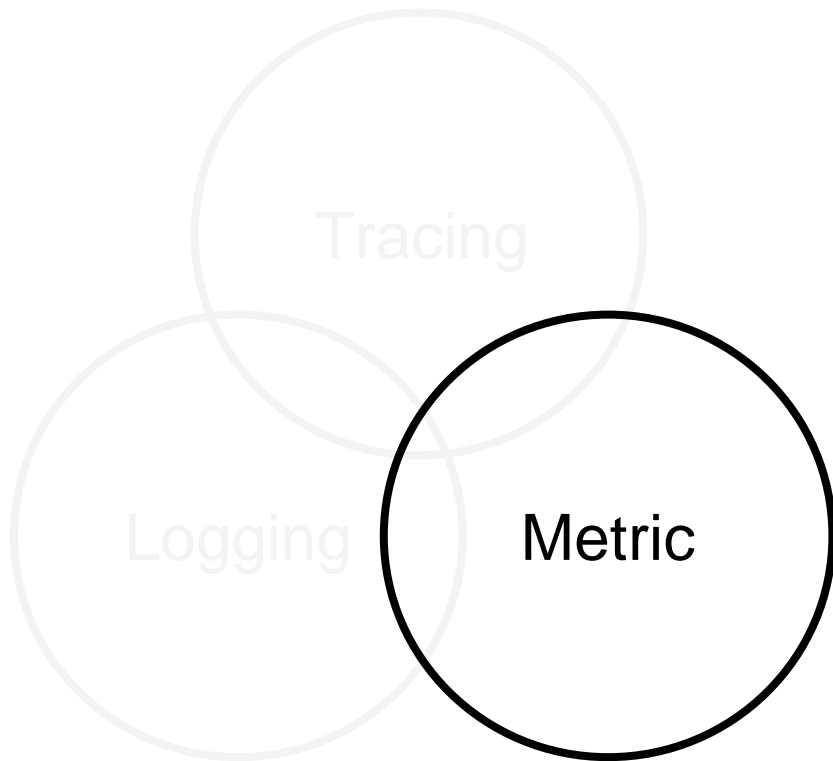
그리고 사람

## Observability



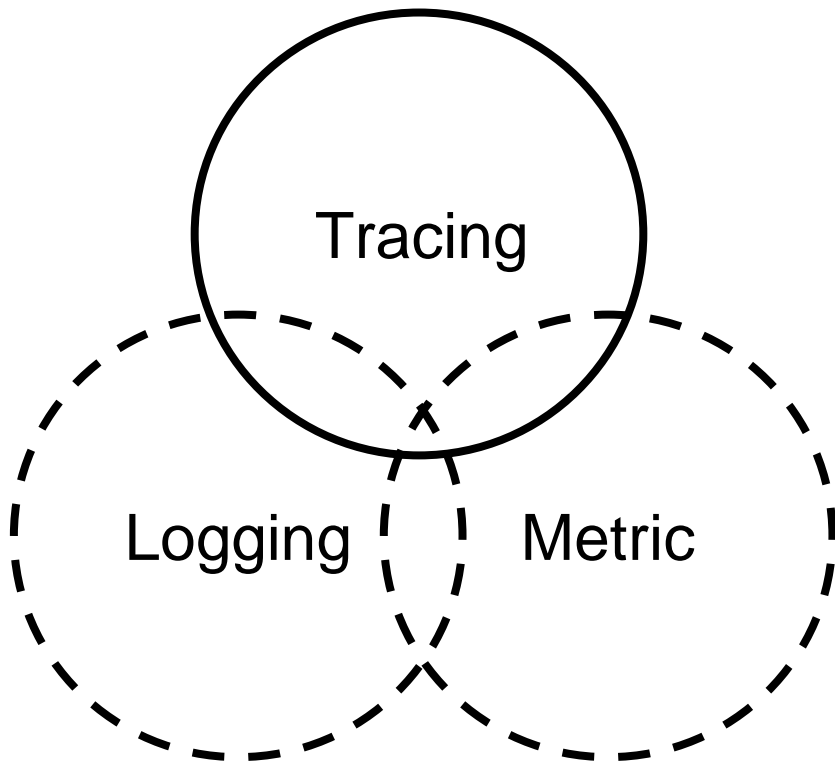
NELO2

## Observability



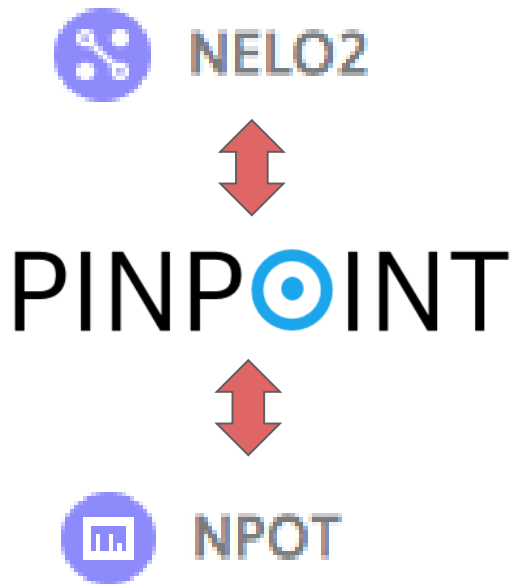
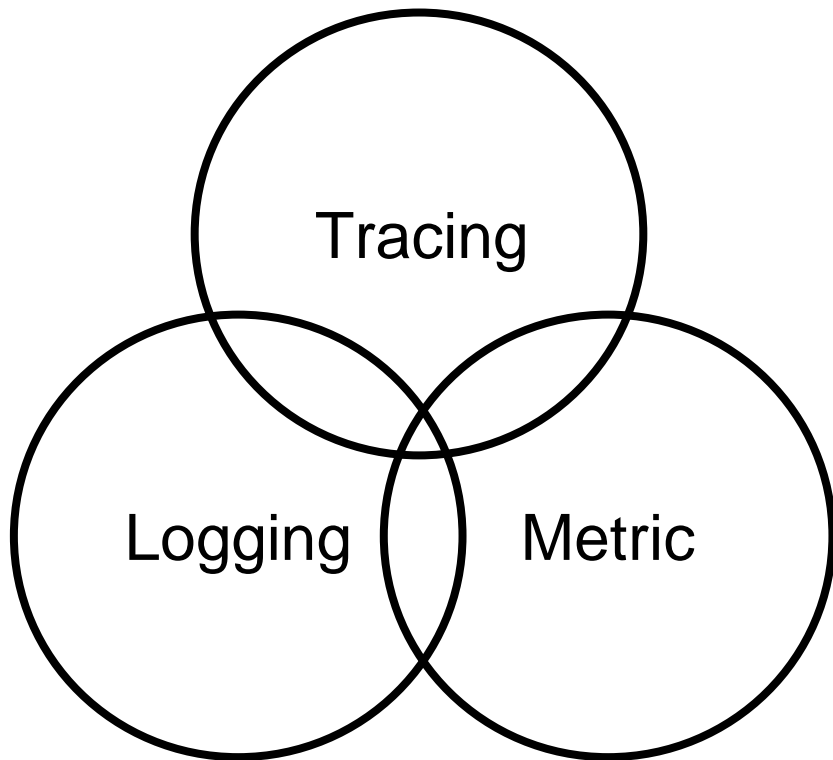
NPOT

## Observability

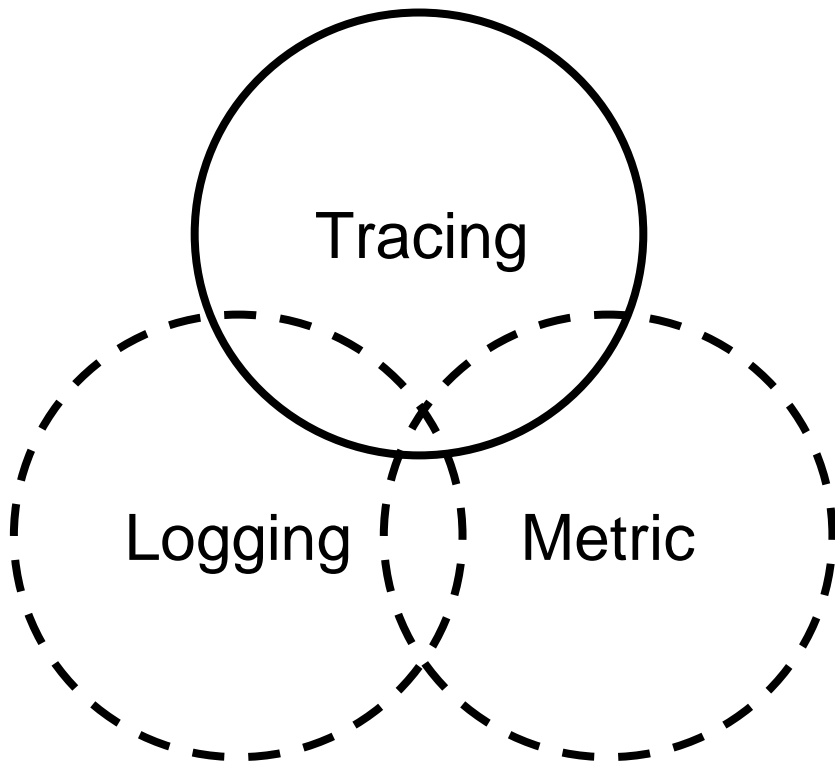


PINPOINT

## Observability



## Observability



PINPOINT

PINPOINT

## Bird Eye View

Finding Slow Transactions

Distributed Tracing

DevOps

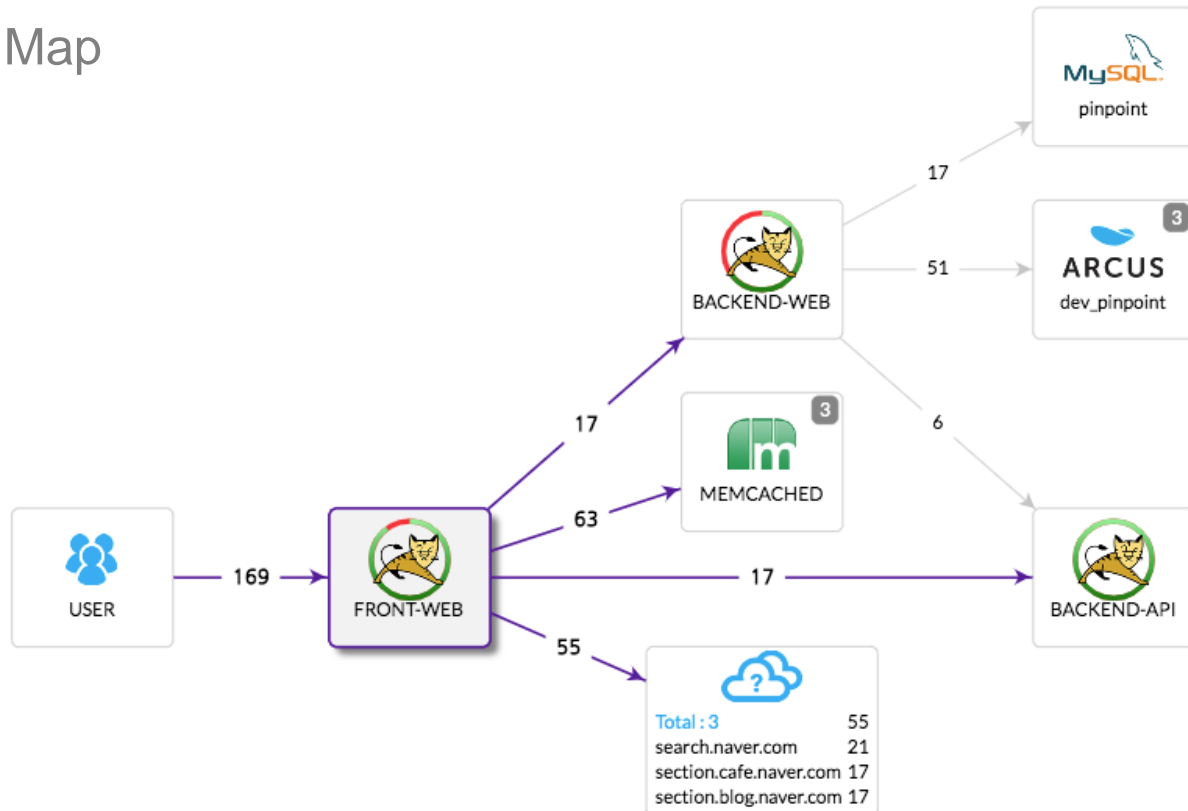
Scalable

Minimum Overload



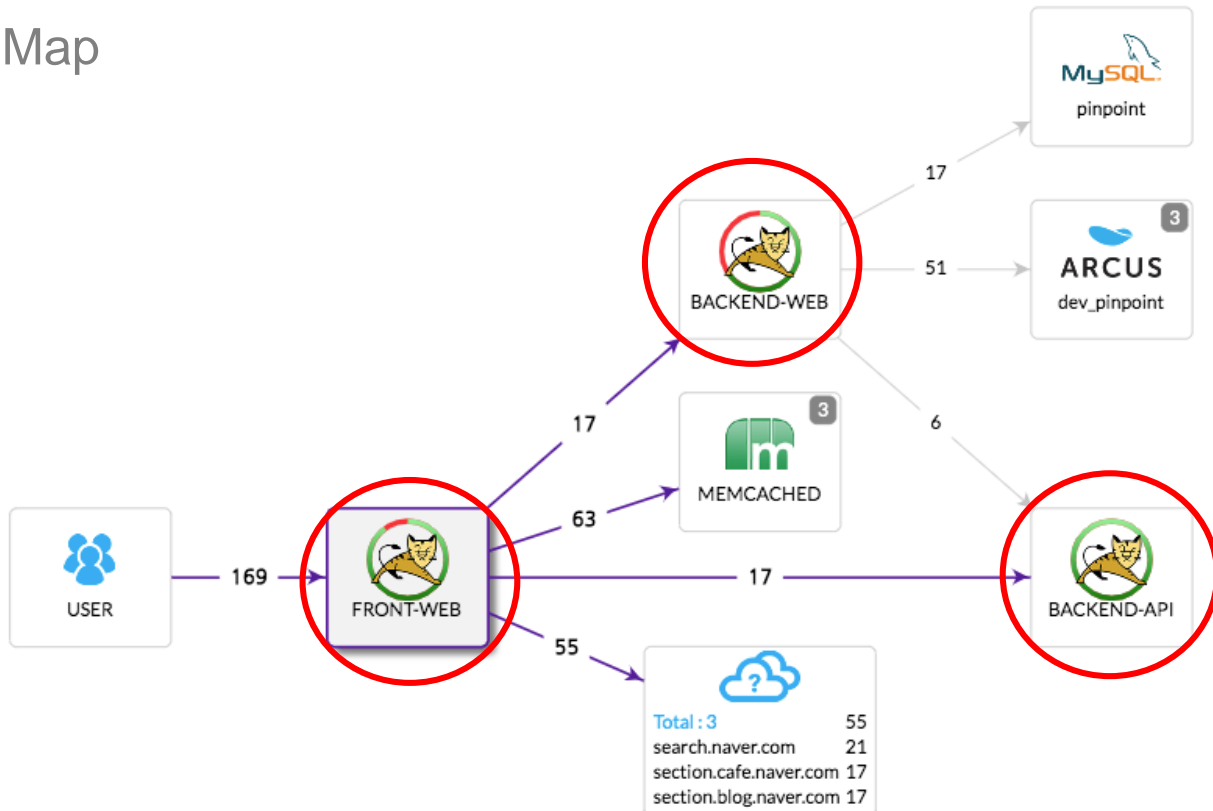
# Bird Eye View

- Server Map



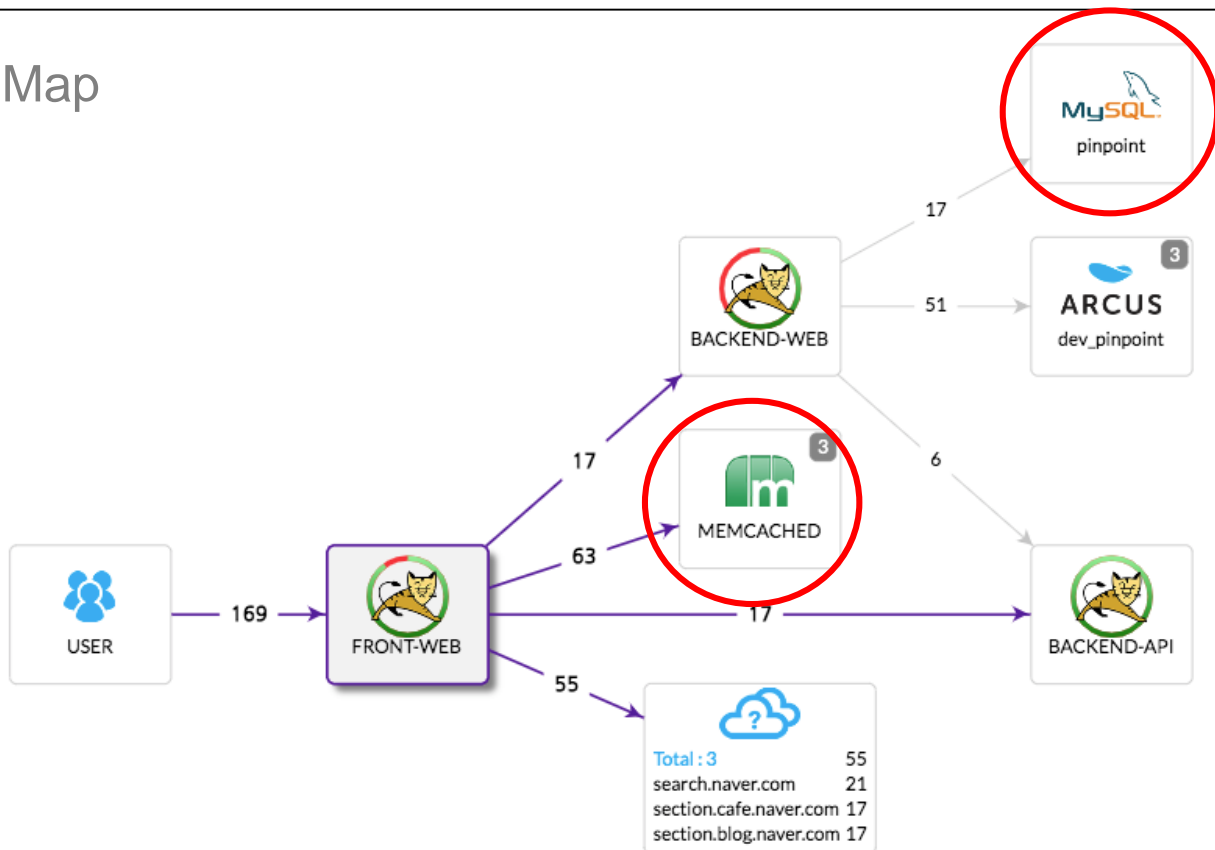
# Bird Eye View

- Server Map



# Bird Eye View

- Server Map

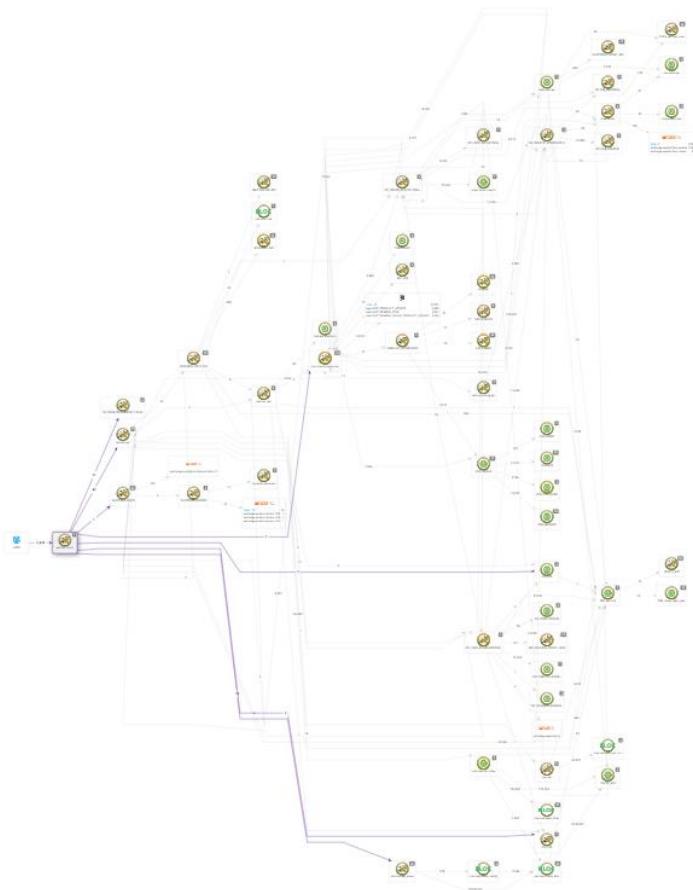


# Bird Eye View



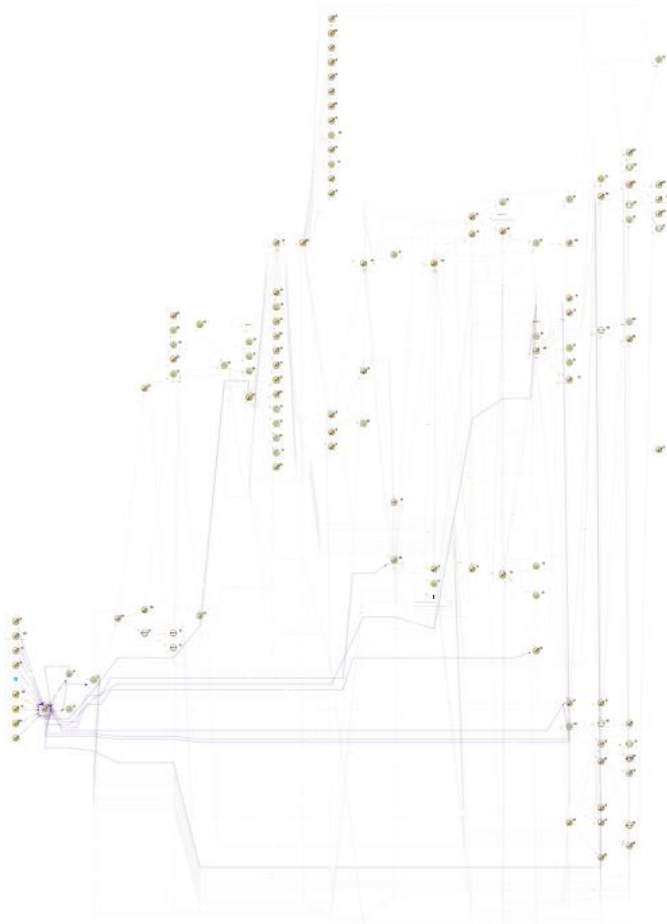
# Bird Eye View

---



# Bird Eye View

---



PINPOINT

Bird Eye View

**Finding Slow Transactions**

Distributed Tracing

DevOps

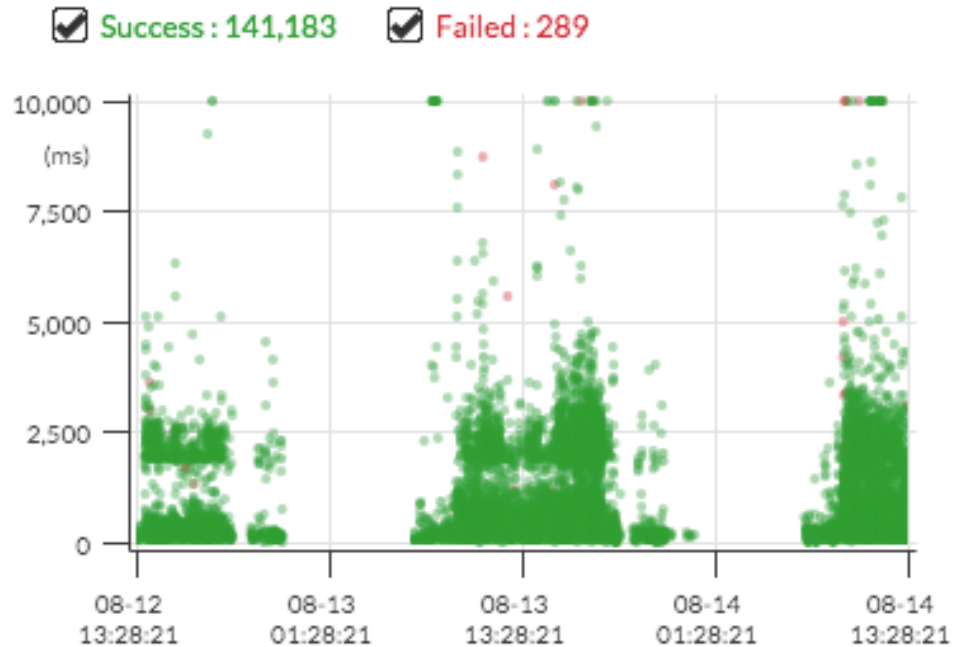
Scalable

Minimum Overload

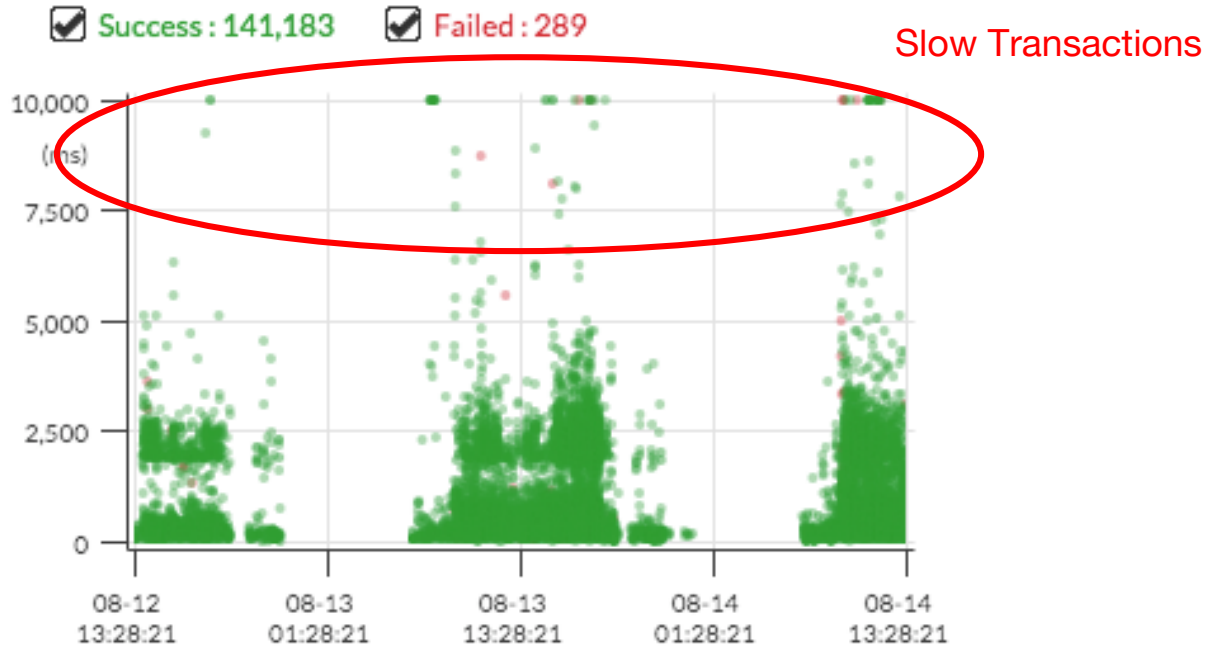




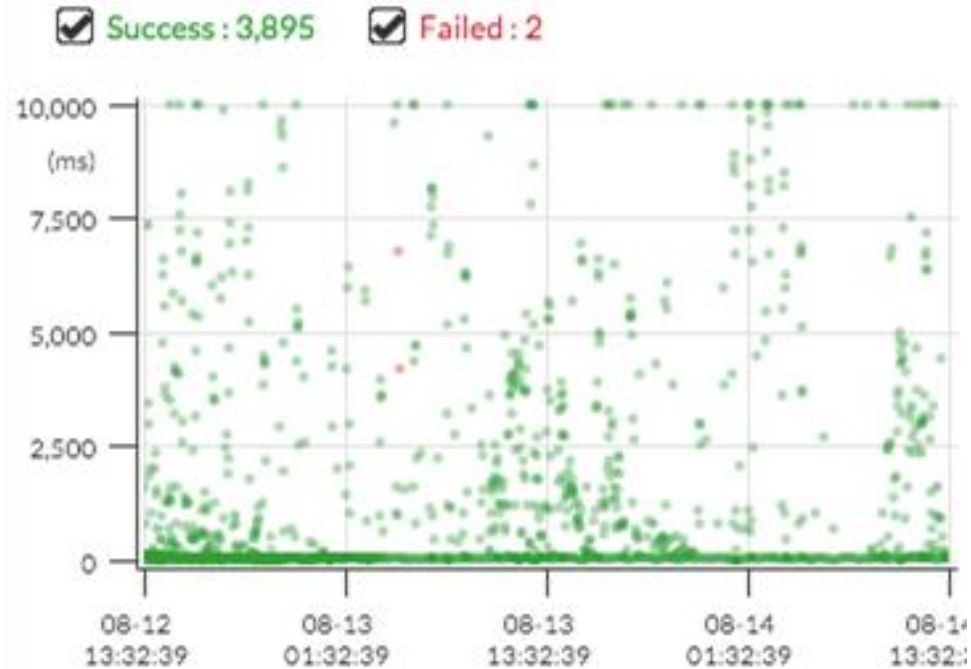
# Finding Slow Transaction



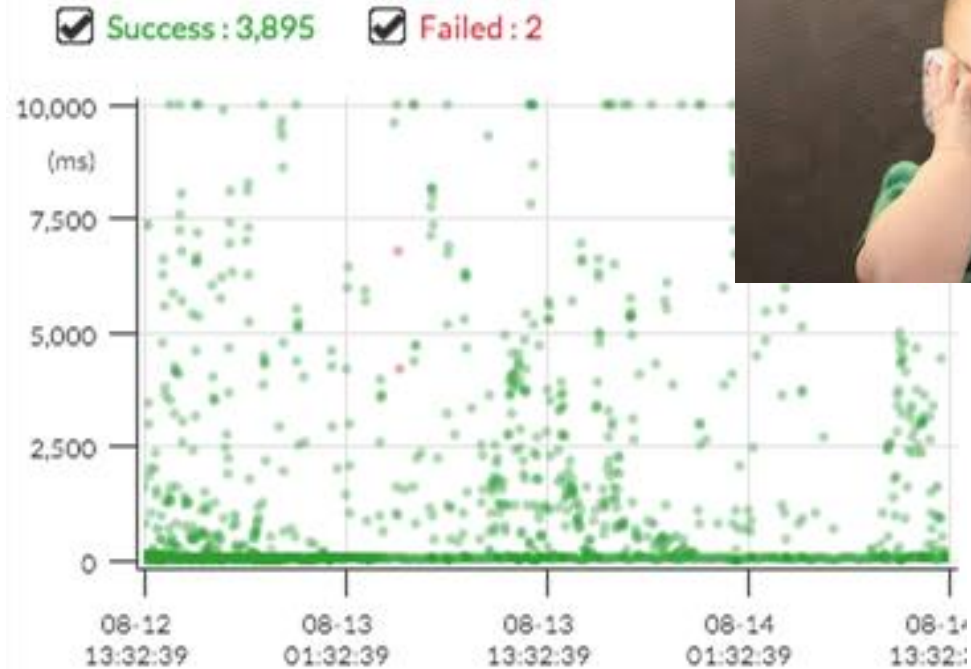
# Finding Slow Transaction



# Finding Slow Transaction



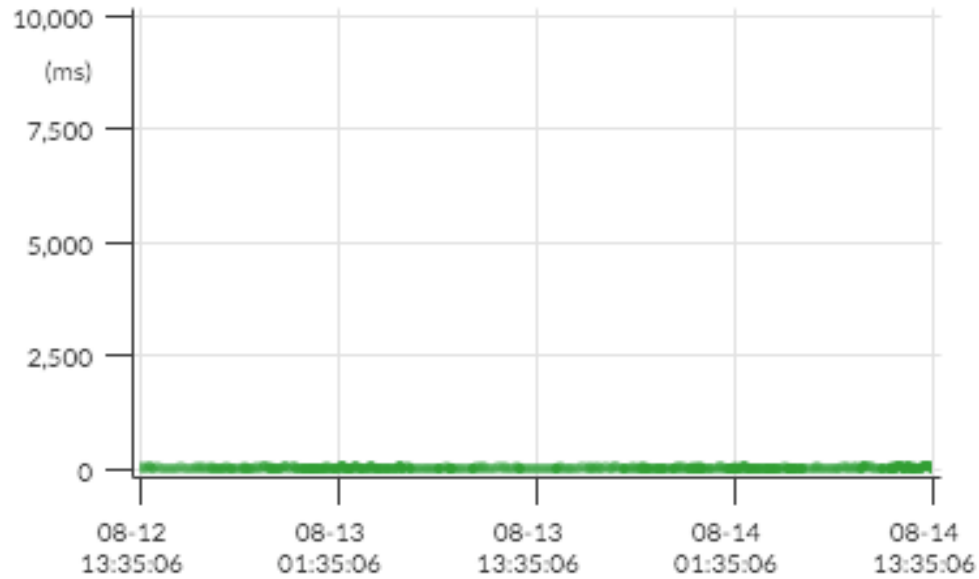
# Finding Slow Transaction



# Finding Slow Transaction

---

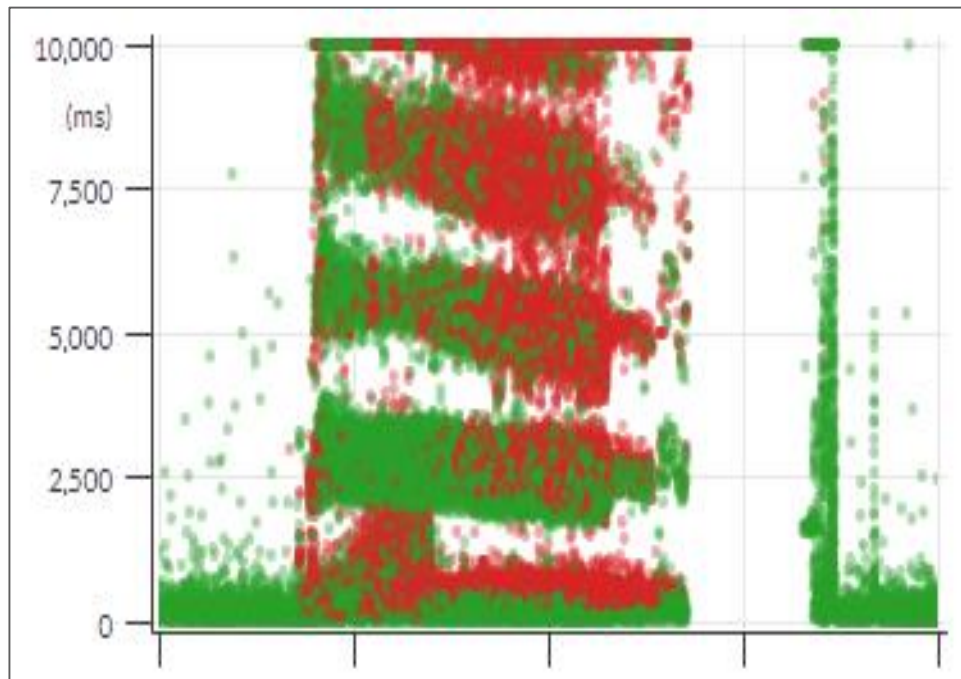
Success : 4,906     Failed : 0



# Finding Slow Transaction

---

- Malfunctioning in Real Server



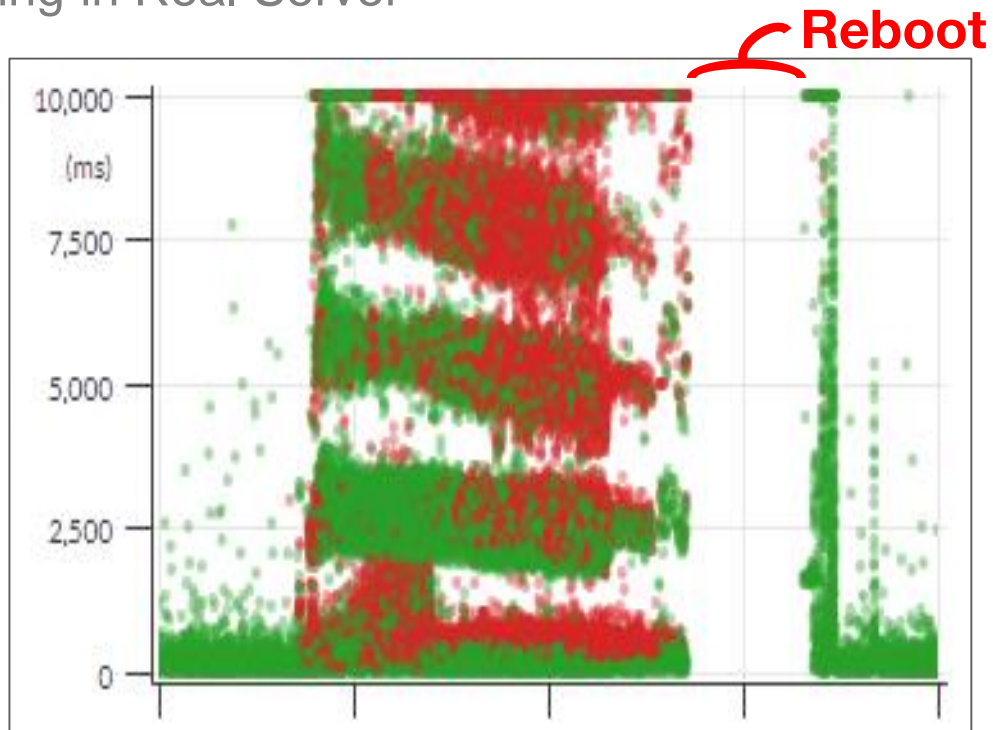
# Finding Slow Transaction

- Malfunctioning in Real Server



# Finding Slow Transaction

- Malfunctioning in Real Server

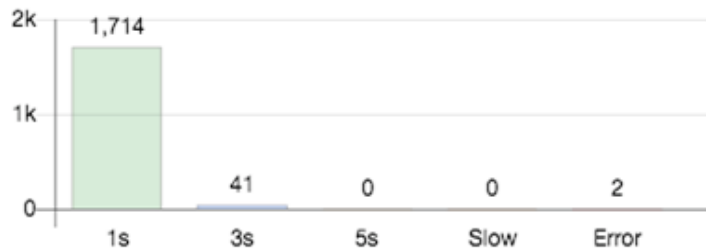




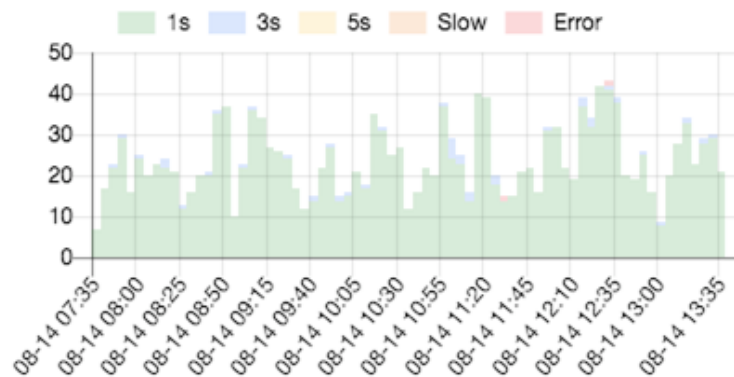
# Finding Slow Transaction

- Response Summary chart
- Load chart

Response Summary



Load



# Enhance Observability

---

PINPOINT

Bird Eye View

Finding Slow Transactions

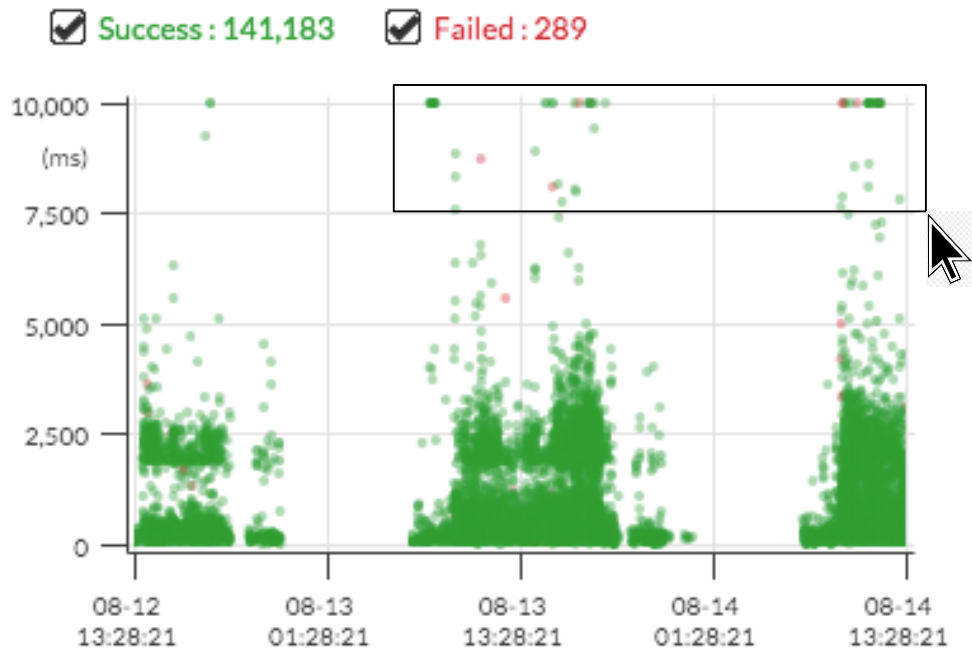
**Distributed Tracing**

DevOps

Scalable

Minimum Overload

# Finding Slow Transaction





# Distributed Tracing

**PINPOINT** 12:22 12:22 12:22 12:23 12:23 12:23 12:23 12:23 12:23 Done (7/7)

#	Start Time	Path	Res. (ms) ↓	Exception	Agent	Client IP	Transaction
2	08/17 12:23:32.480	/v1/shopping/orders/1340cbae-b9e8-4a41-a36b-57206ab17443	7,087		apigw01	127.0.0.1	apigw01^1565160016090^3378633
4	08/17 12:23:16.925	/v1/shopping/orders/6ac3cb68-23c8-4b76-955f-4c479672715a	6,968		apigw01	127.0.0.1	apigw01^1565160016090^3378558
3	08/17 12:23:24.881	/v1/shopping/orders/12d2efb-683e-49ec-9443-1b8ddcf022b	6,056		apigw01	127.0.0.1	apigw01^1565160016090^3378577
7	08/17 12:22:59.601	/v1/shopping/orders/24b01dfe-0345-4c84-995a-c8e392136ef8	5,867		apigw01	127.0.0.1	apigw01^1565160016090^3378478
5	08/17 12:23:14.958	/v1/shopping/orders/674ceb8-0b41-4caa-8212-b84124cb94fd	5,170		apigw01	127.0.0.1	apigw01^1565160016090^3378545
1	08/17 12:23:36.629	/v1/shopping/orders/b6dea01f-3063-4bd3-bece-69f9c543019	5,039		apigw01	127.0.0.1	apigw01^1565160016090^3378641
6	08/17 12:23:03.896	/v1/shopping/orders/4eeef04c-70c9-4379-90ae-5f76715493c1	4,929		apigw01	127.0.0.1	apigw01^1565160016090^3378480

Application: /v1/shopping/orders/1340cbae-b9e8-4a41-a36b-57... TransactionId: apigw01^1565160016090^3378633 AgentId: apigw01 ApplicationName: ApiGateway

Call Tree Server Map Timeline Mixed View Complete

GoJS evaluation  
(c) 1998-2016 Northwoods Software  
Not for distribution or production use  
nwoods.com

```
graph LR; USER[USER] -- 1 --> ApiGateway[ApiGateway]; ApiGateway -- 1 --> ShoppingApi[Shopping-Api]; ShoppingApi -- 1 --> ShoppingOrder[Shopping-Order]; ShoppingOrder -- 1 --> ShoppingPayment[Shopping-Payment]; ShoppingOrder -- 3 --> MySQL[MySQL order]; ShoppingPayment -- 1 --> VendorPayment[vendor.payment.card.com:8179];
```

# Distributed Tracing

**PINPOINT** 12:22 12:22 12:22 12:23 12:23 12:23 12:23 12:23 12:23 Done (7/7)

#	Start Time	Path	Res. (ms) ↓	Exception	Agent	Client IP	Transaction
2	08/17 12:23:32.480	/v1/shopping/orders/1340cbac-b9e8-4a41-a36b-57206ab17443	7,087		apigw01	127.0.0.1	apigw01^1565160016090^3378633
4	08/17 12:23:16.925	/v1/shopping/orders/6ac3cb68-23c8-4b76-955f-4c479672715a	6,968		apigw01	127.0.0.1	apigw01^1565160016090^3378558
3	08/17 12:23:24.881	/v1/shopping/orders/12dcef7b-683e-49ec-9443-1b8ddcf022b	6,056		apigw01	127.0.0.1	apigw01^1565160016090^3378577
7	08/17 12:22:59.601	/v1/shopping/orders/24b01dfe-0345-4c84-995a-c8e392136ef8	5,867		apigw01	127.0.0.1	apigw01^1565160016090^3378478
5	08/17 12:23:14.958	/v1/shopping/orders/674ceb78-0b41-4caa-8212-b84124cb94fd	5,170		apigw01	127.0.0.1	apigw01^1565160016090^3378545
1	08/17 12:23:36.629	/v1/shopping/orders/b6dea01f-3063-4bd3-bece-69f9c543019	5,039		apigw01	127.0.0.1	apigw01^1565160016090^3378641
6	08/17 12:23:03.896	/v1/shopping/orders/4eeef04c-70c9-4379-90ae-5f76715493c1	4,929		apigw01	127.0.0.1	apigw01^1565160016090^3378480

Application: /v1/shopping/orders/1340cbac-b9e8-4a41-a36b-57... TransactionId: apigw01^1565160016090^3378633 AgentId: apigw01 ApplicationName: ApiGateway

Call Tree Server Map Timeline Mixed View Complete

GoJS evaluation  
(c) 1998-2016 Northwoods Software  
Not for distribution or production use  
nwoods.com

```
graph LR; USER[USER] -- 1 --> ApiGateway[ApiGateway]; ApiGateway -- 1 --> ShoppingApi[Shopping-API]; ShoppingApi -- 1 --> ShoppingOrder[Shopping-Order]; ShoppingOrder -- 1 --> ShoppingPayment[Shopping-Payment]; ShoppingOrder -- 3 --> MySQL[MySQL order]; ShoppingPayment -- 1 --> VendorPayment[vendor.payment.card.com:8179];
```

The diagram illustrates a distributed tracing call tree. It starts with a 'USER' icon, followed by a red circle around the '1' on the arrow pointing to 'ApiGateway'. From 'ApiGateway', an arrow with '1' points to 'Shopping-API', which then points to 'Shopping-Order' (also with '1'). From 'Shopping-Order', an arrow with '1' points to 'Shopping-Payment', and another arrow with '3' points to a 'MySQL order' box. Finally, an arrow with '1' points from 'Shopping-Payment' to a 'vendor.payment.card.com:8179' box.



# Enhance Observability

---

PINPOINT

Bird Eye View

Finding Slow Transactions

Distributed Tracing

**DevOps**

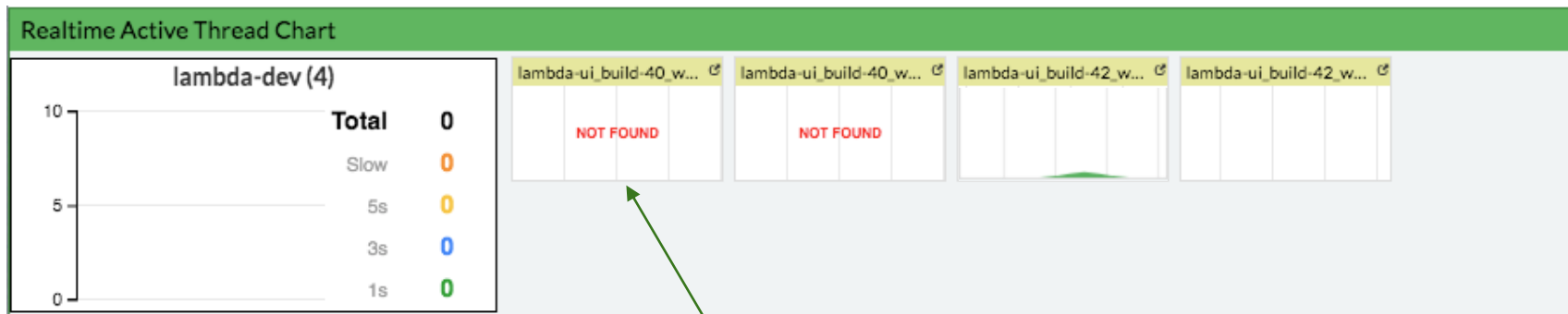
Scalable

Minimum Overload



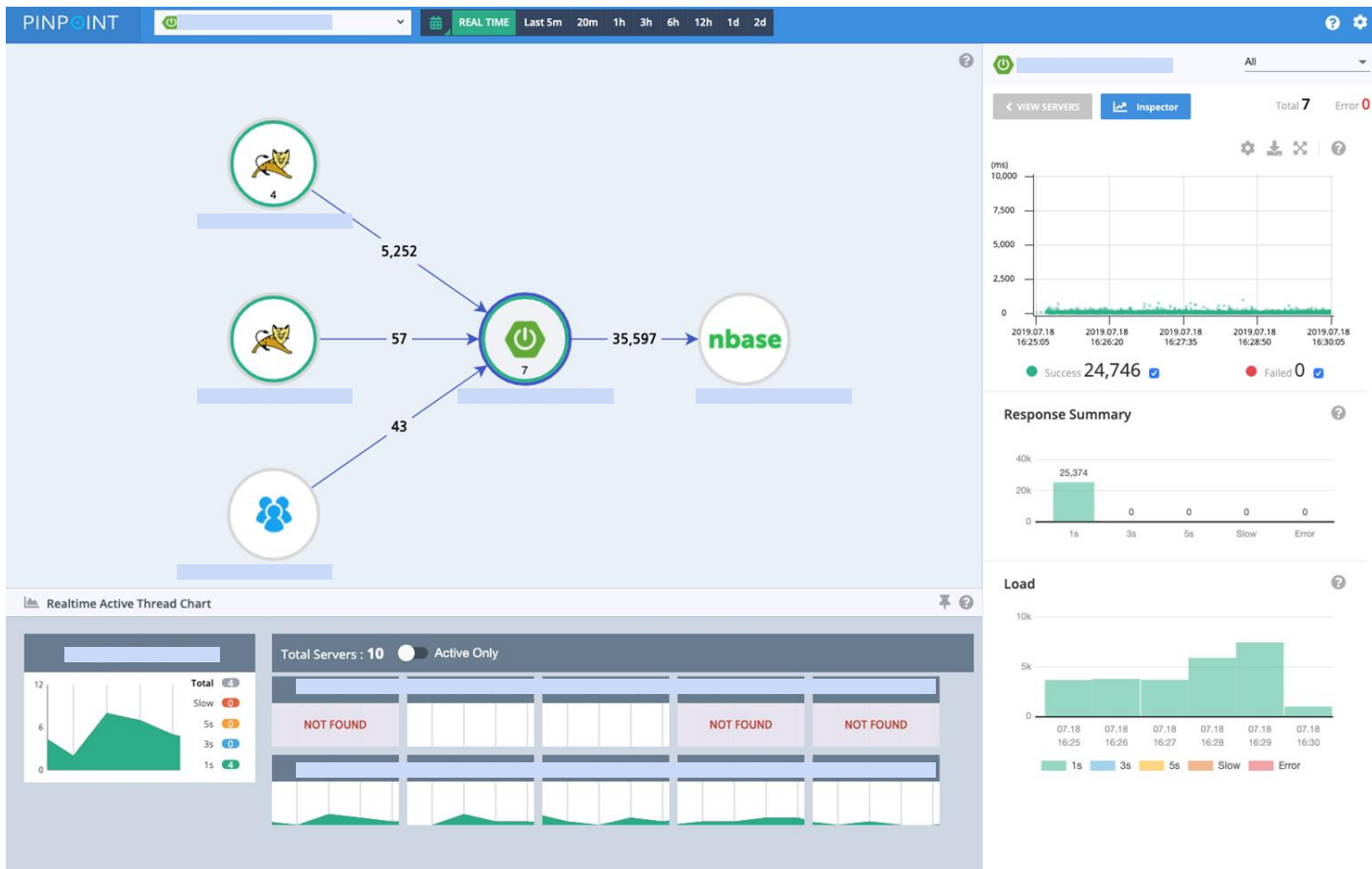
# DevOps

- Realtime Active Thread Chart

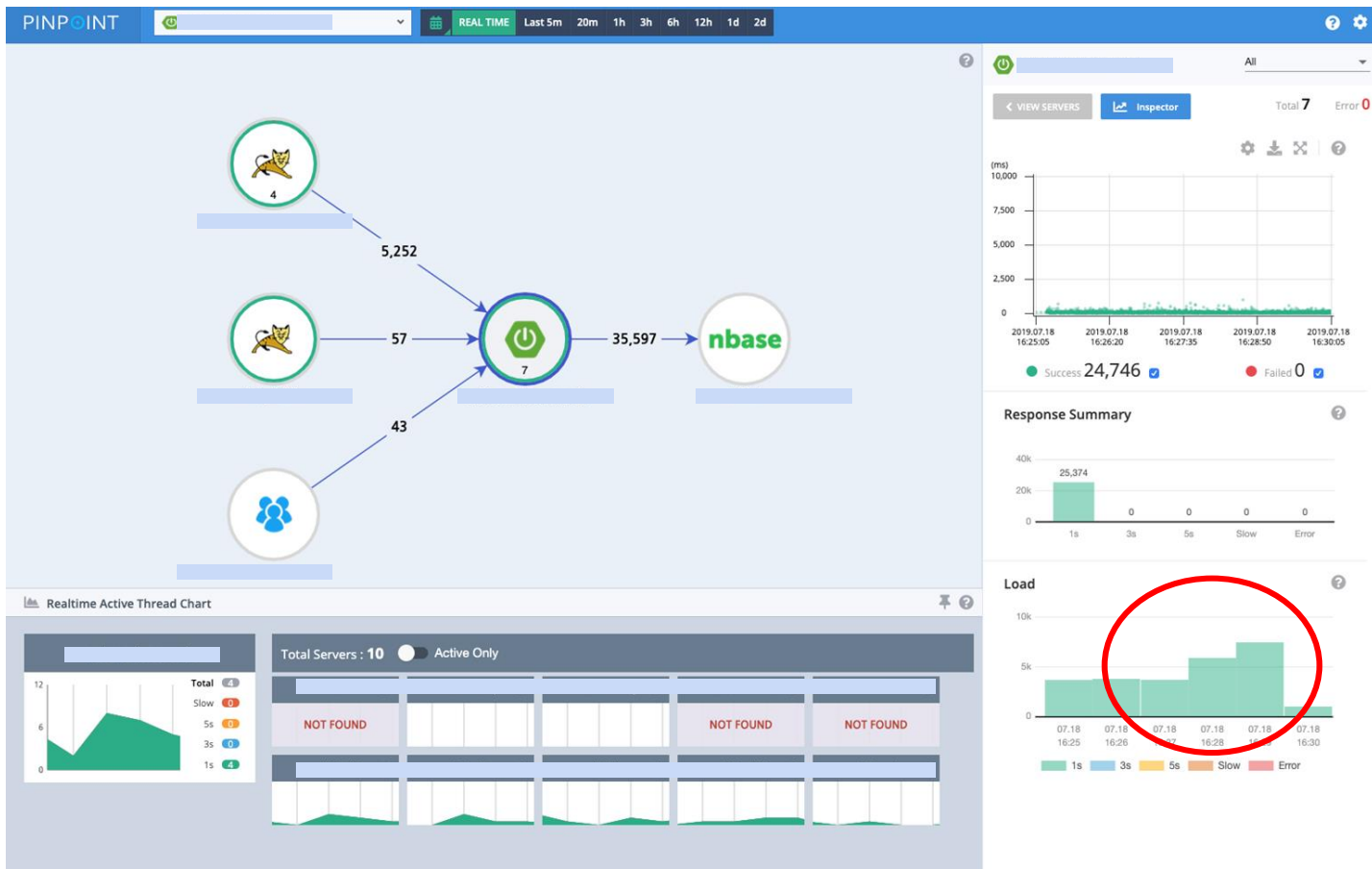


Can be used as HealthCheck

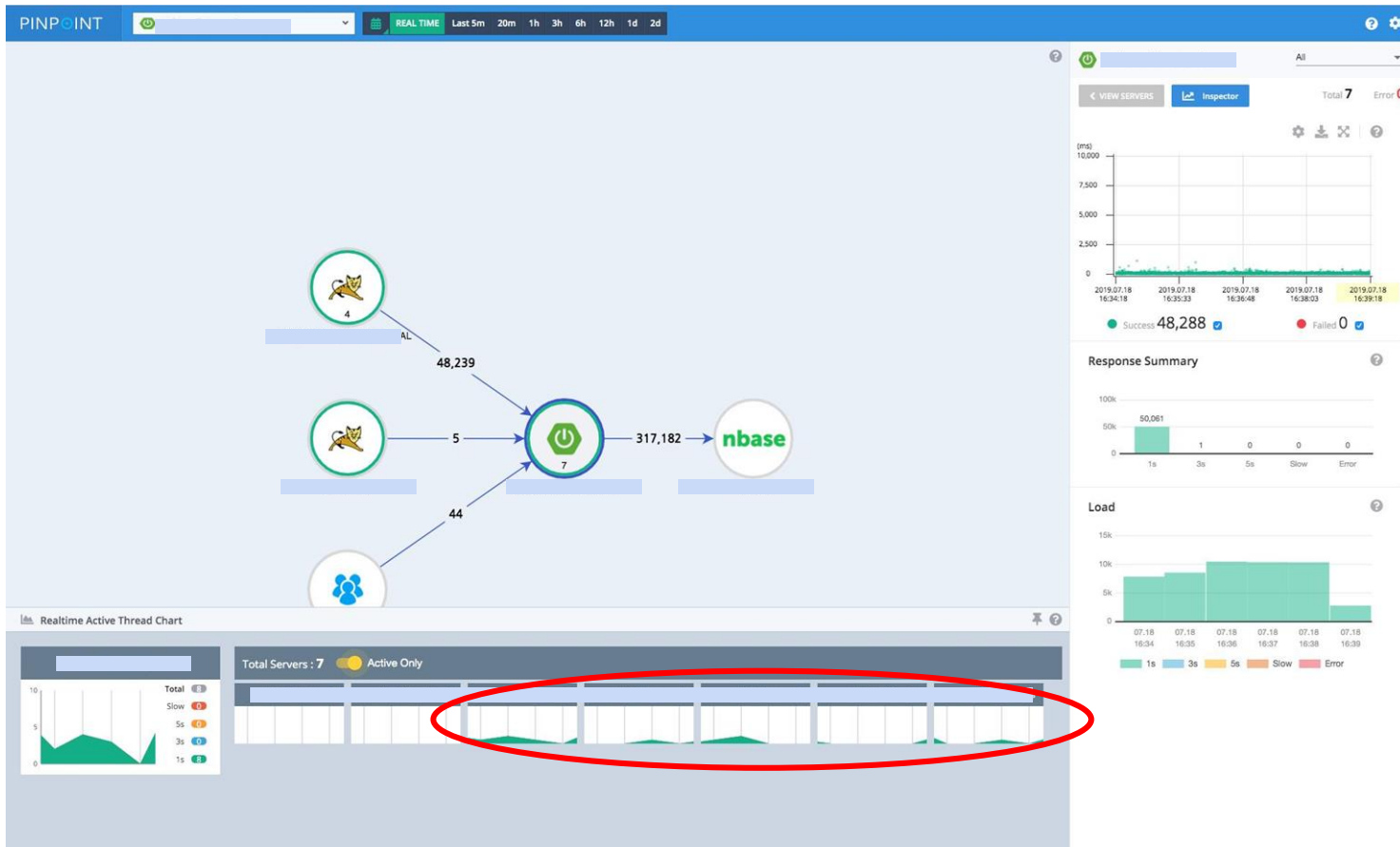
# DevOps



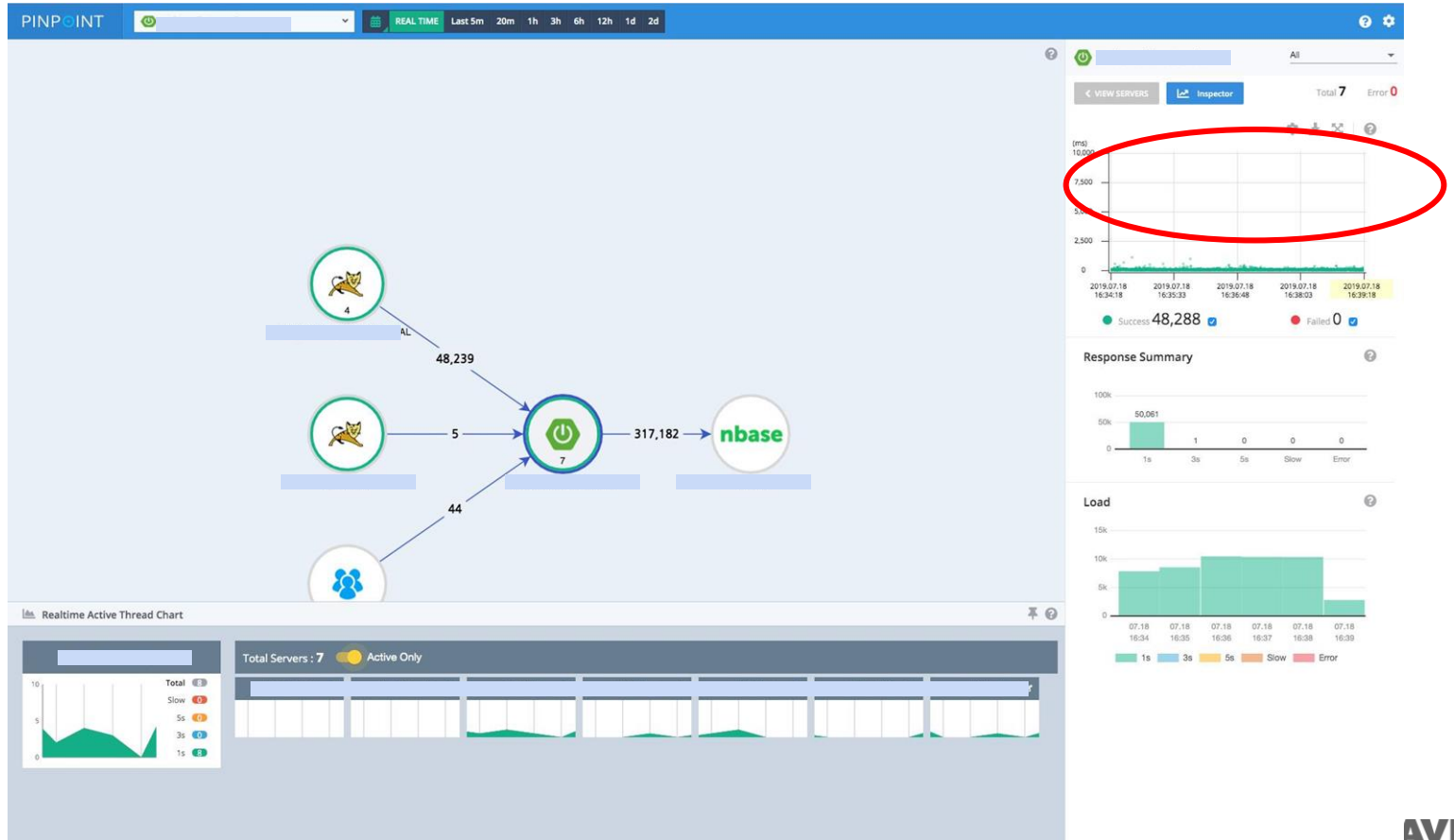
# DevOps



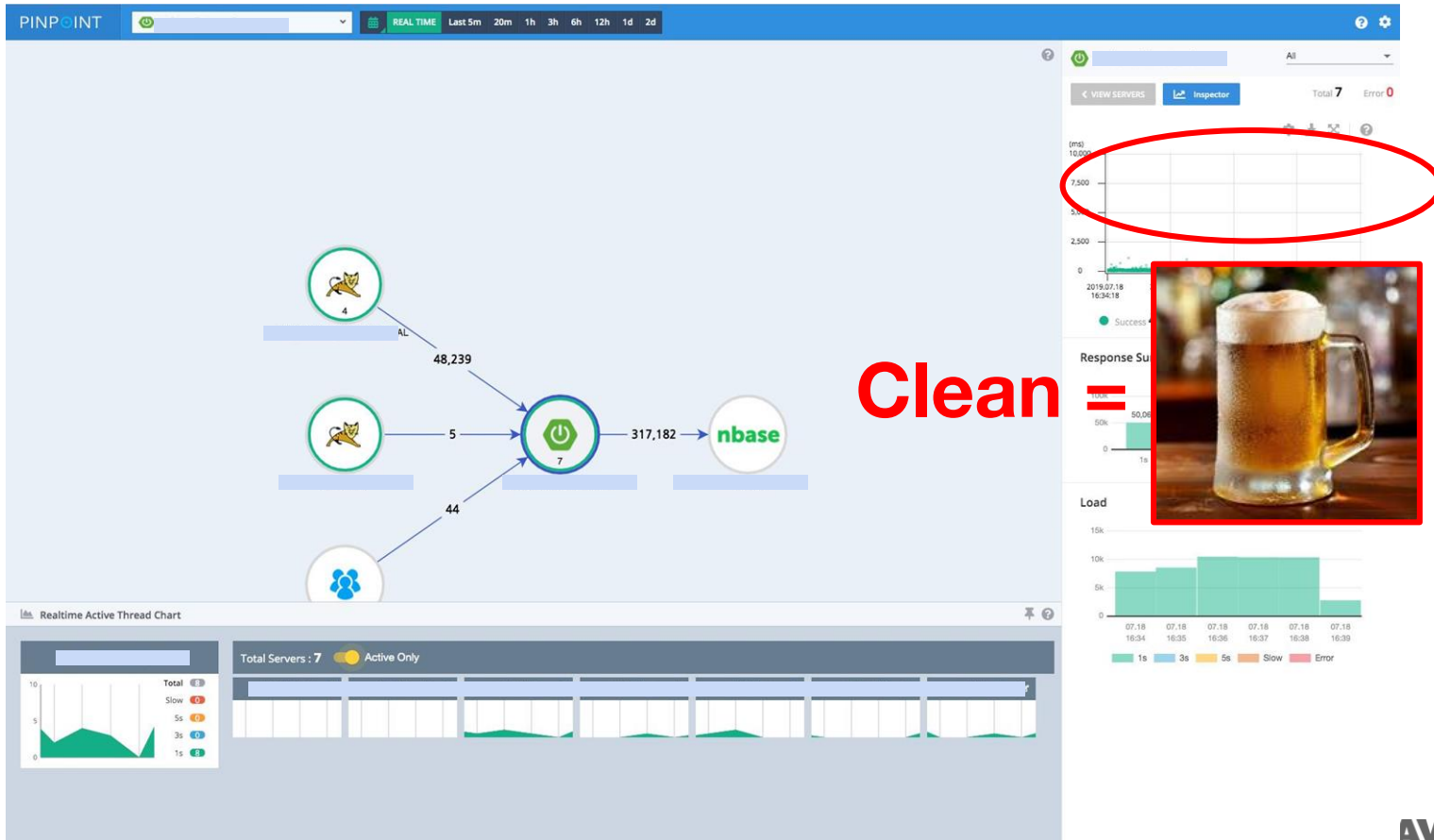
# DevOps



# DevOps



# DevOps



# DevOps

- Inspector

## Basic Info of the Instance



- Heap, Non-Heap Memory
- JVM/SYSTEM CPU
- JVM GC
- TPS, Active Thread
- Response Time
- File Descriptor
- Direct/Mapped Buffer
- Data Source

# Enhance Observability

---

PINPOINT

Bird Eye View

Finding Slow Transactions

Distributed Tracing

DevOps

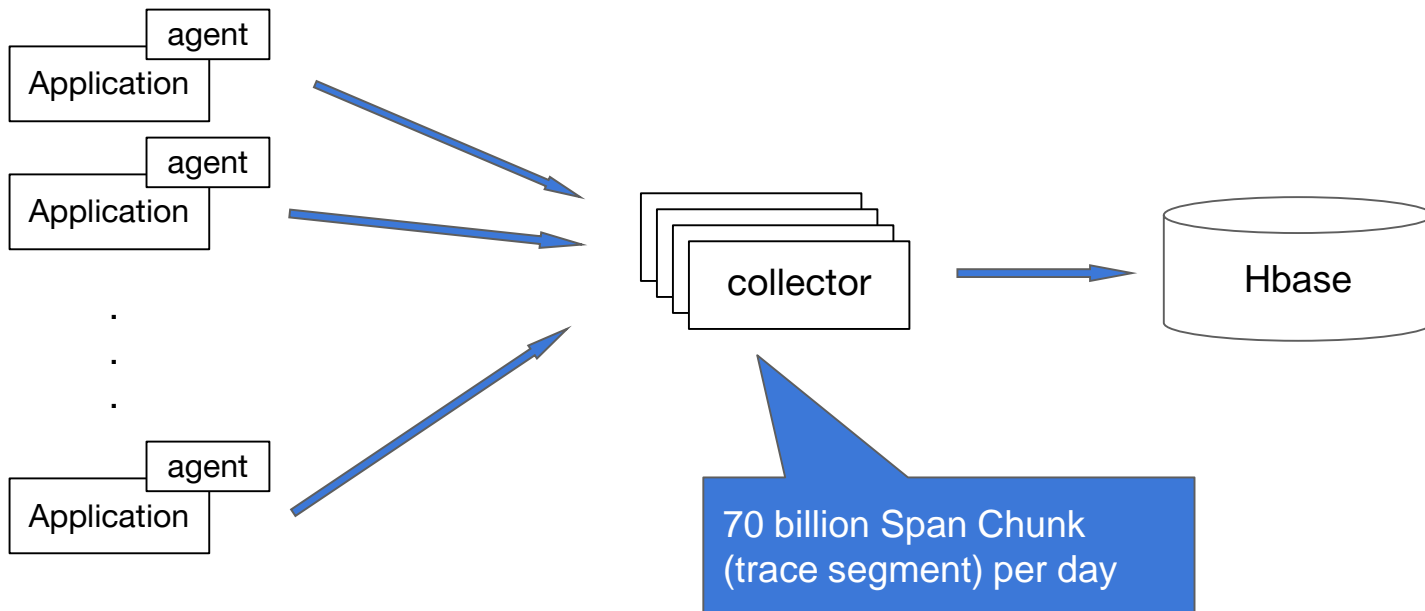
**Scalable**

Minimum Overload



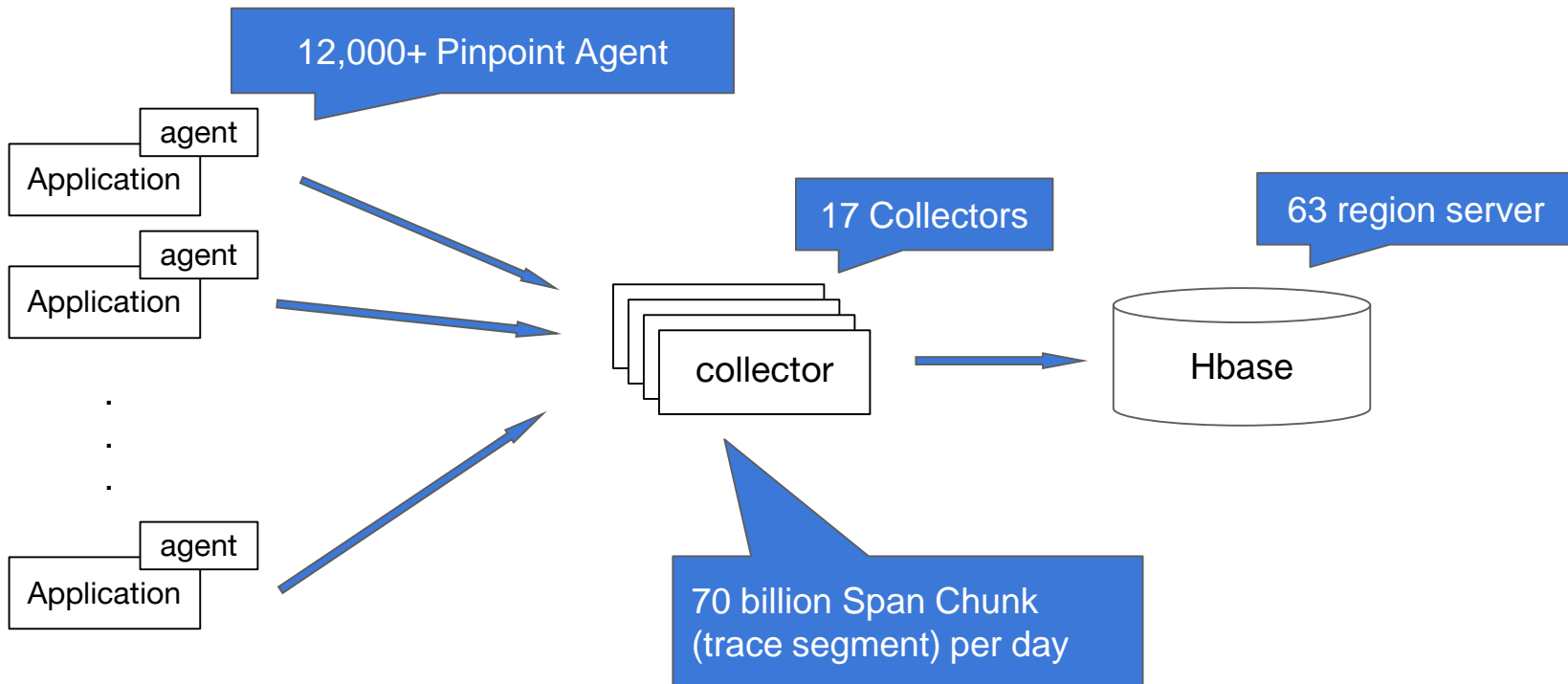
# Scalable

- Fully functioning in Naver



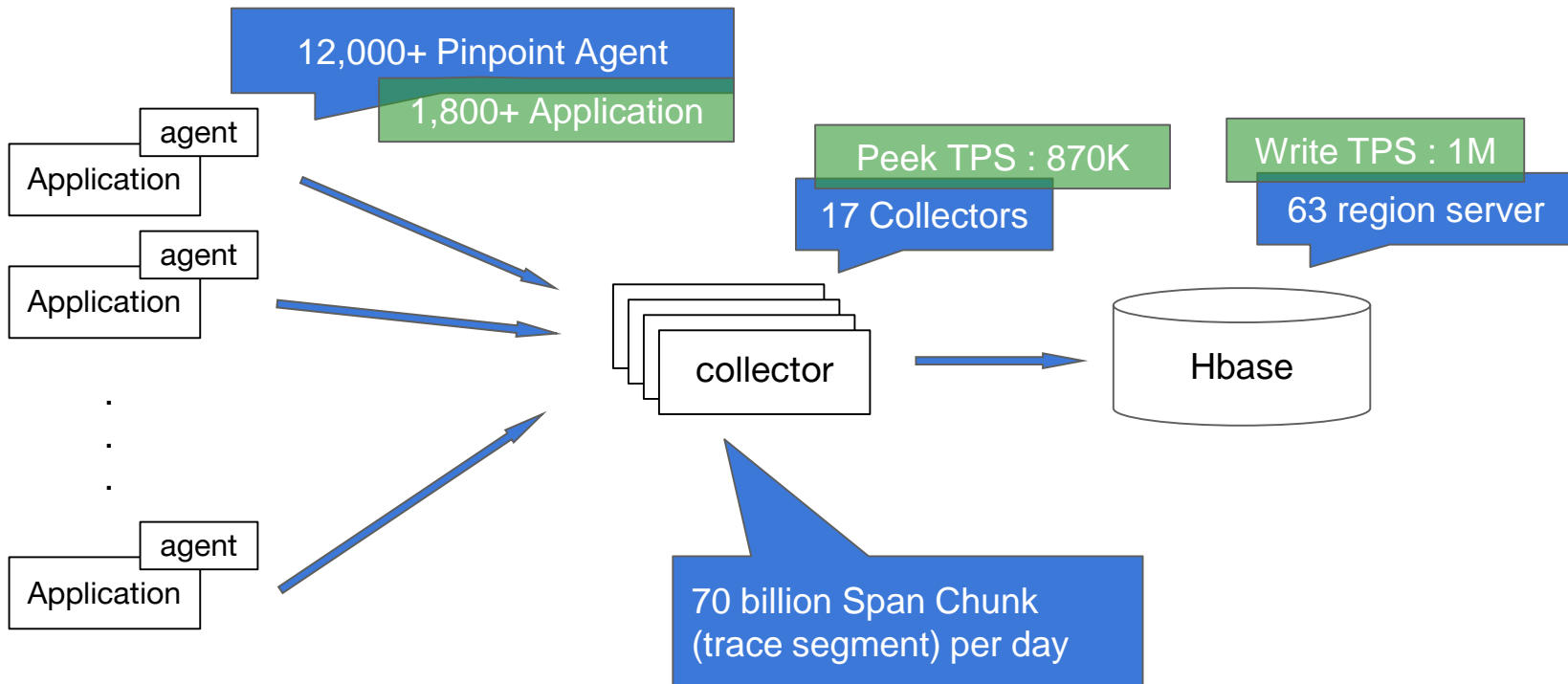
# Scalable

- Fully functioning in Naver



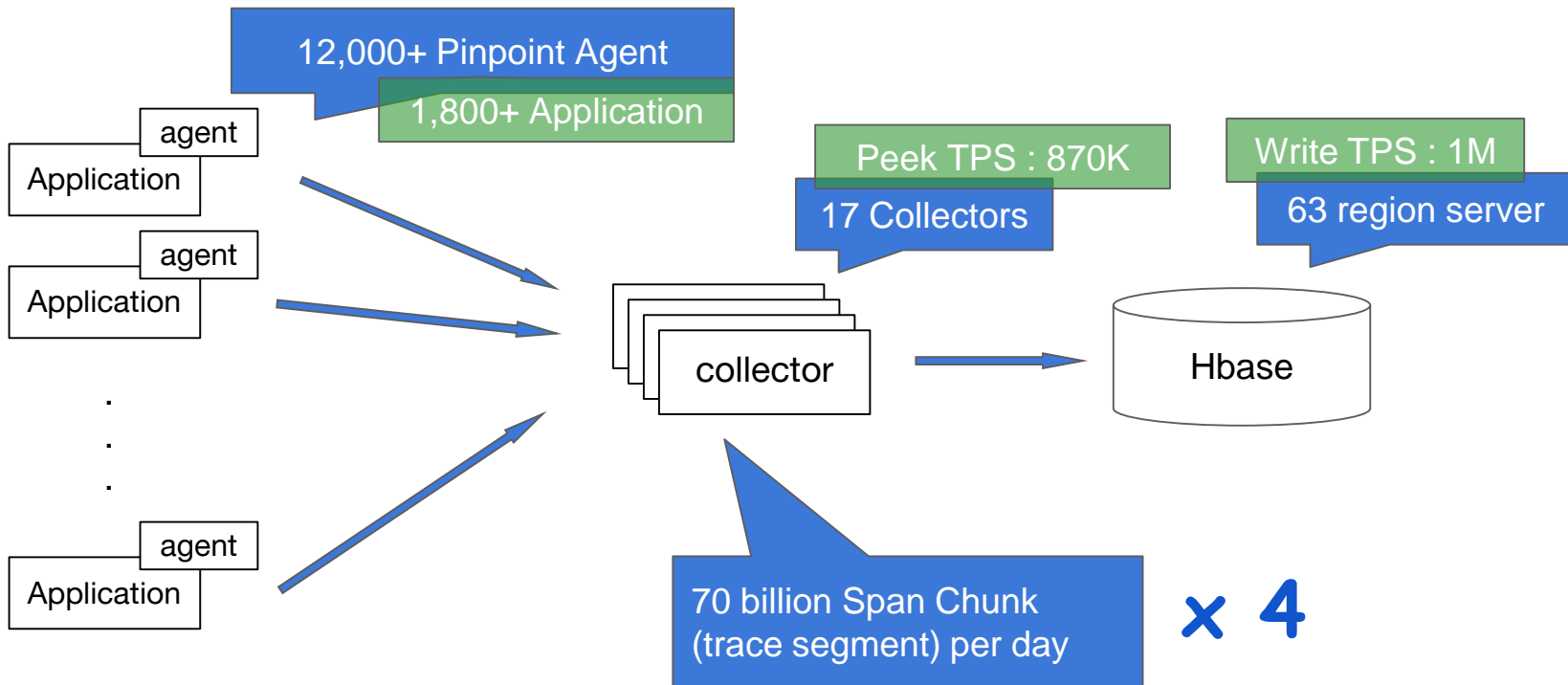
# Scalable

- Fully functioning in Naver



# Scalable

- Fully functioning in Naver



# Enhance Observability

---

PINPOINT

Bird Eye View

Finding Slow Transactions

Distributed Tracing

DevOps

Scalable

**Minimum Overload**

# Minimum Overload

- Execute Integration Tests Periodically
  - **Less than 3%** difference in performance
  - Sampling 'No-Agent', '5%', '100%'

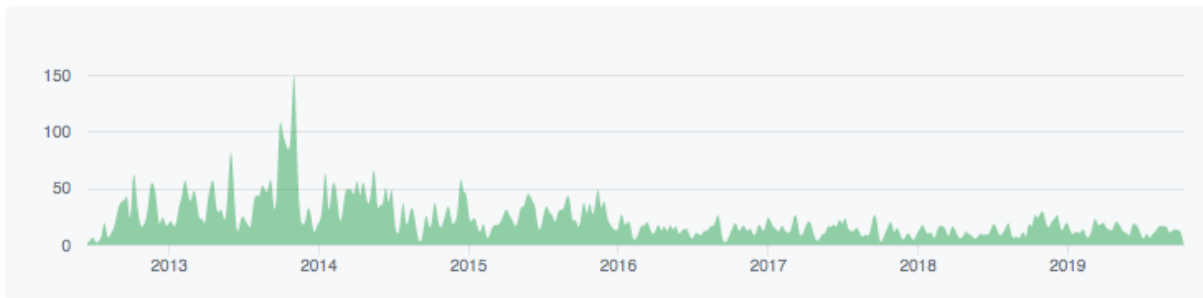
Tomcat Only																							
VUser 4						VUser 32						VUser 64						VUser 128					
Duration (ms)	Tests	Error	Mean Test Time (ms)	TPS	nGrinder	Duration (ms)	Tests	Error	Mean Test Time (ms)	TPS	nGrinder	Duration (ms)	Tests	Error	Mean Test Time (ms)	TPS	nGrinder	Duration (ms)	Tests	Error	Mean Test Time (ms)	TPS	nGrinder
300000	135003	0	6.87	571.92	<a href="#">link</a>	300000	421252	20	17.63	1784.54	<a href="#">link</a>	300000	427943	11	34.78	1828.37	<a href="#">link</a>	300000	427923	0	69.95	1828.08	<a href="#">link</a>
300000	125282	0	7.42	530.74	<a href="#">link</a>	300000	422774	23	17.51	1791.0	<a href="#">link</a>	300000	415028	7	35.84	1773.2	<a href="#">link</a>	300000	424081	0	70.61	1811.98	<a href="#">link</a>
300000	127462	0	7.29	539.97	<a href="#">link</a>	300000	427575	21	17.34	1811.3	<a href="#">link</a>	300000	428504	8	34.75	1830.7	<a href="#">link</a>	300000	403144	656	72.5	1722.34	<a href="#">link</a>

# Open Source Pinpoint

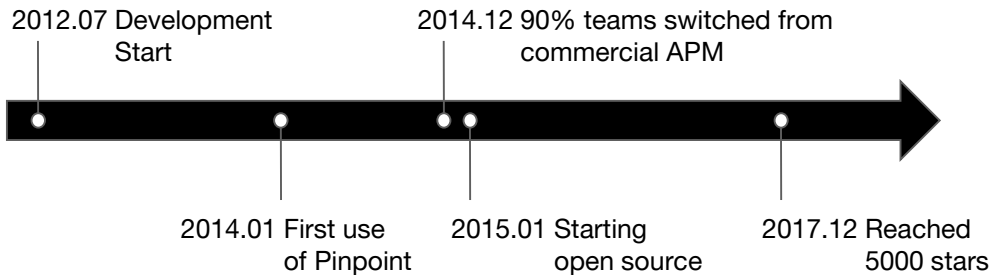
Jun 17, 2012 – Oct 14, 2019

Contributions: Commits ▾

Contributions to master, excluding merge commits



PINPOINT



# Open Source Pinpoint

naver / pinpoint

Unwatch 788

Unstar 9.4k

Fork 2.9k

<> Code

Issues 100

Pull requests 20

Wiki

Security

Insights

Settings

APM, (Application Performance Management) tool for large-scale distributed systems written in Java.

Edit

<https://naver.github.io/pinpoint/>

apm

monitoring

performance

agent

distributed-tracing

tracing

Manage topics

10,739 commits

8 branches

31 releases

1 environment

71 contributors

Apache-2.0

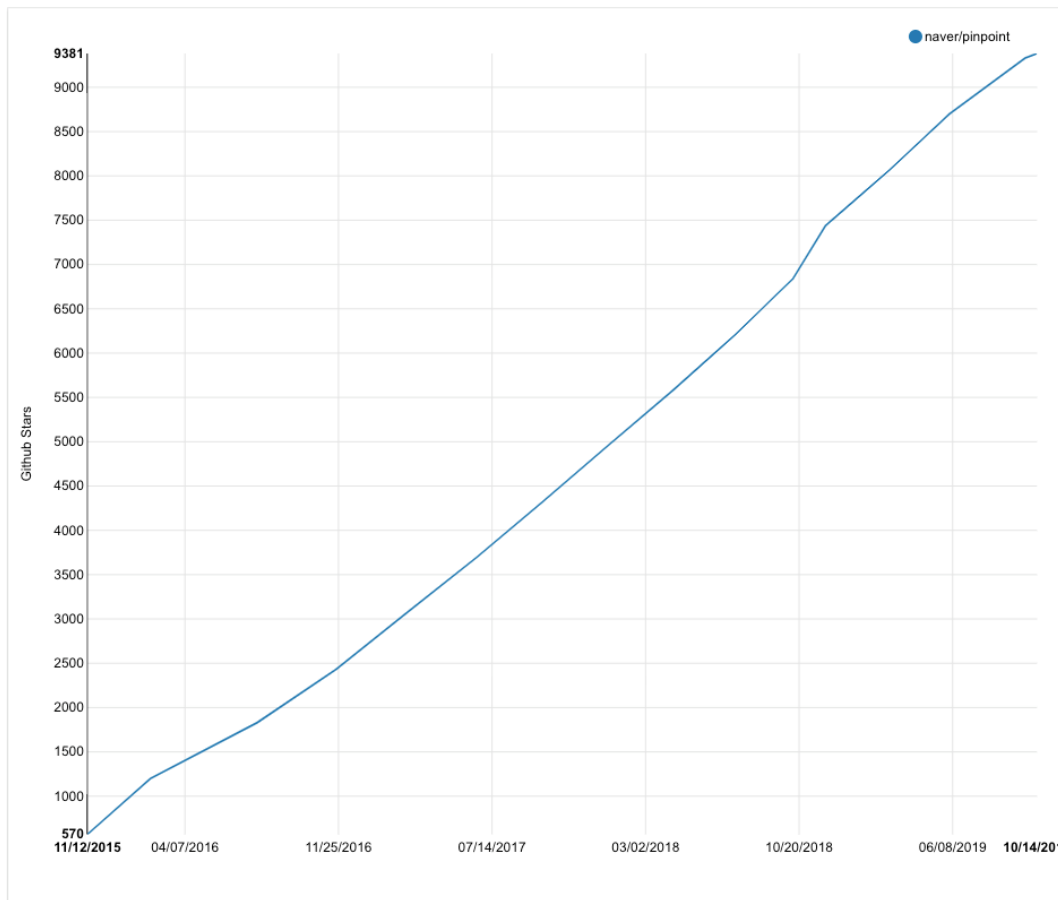
**⚠ We found potential security vulnerabilities in your dependencies.**

You can see this message because you have been granted [access to security alerts for this repository](#).

View security alerts



# Open Source Pinpoint

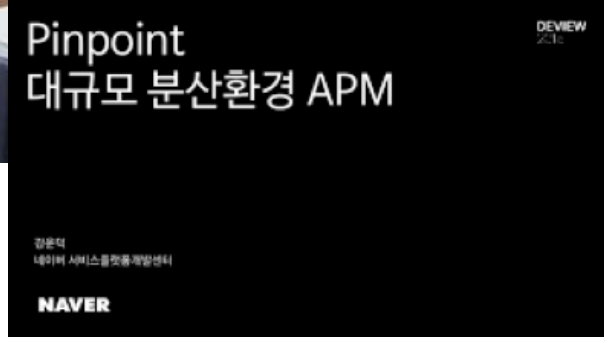


# Open Source Pinpoint



- Lost of Top 10 IT companies in China
- Global enterprises in Korea
- Various IT companies in USA
- Companies in Financial Industry

# Open Source Community



# Open Source Community



# Open Source Community

---





# Open Source Community

---



# Open Source Community



The banner features a vibrant, colorful night view of the Shanghai skyline, including the Oriental Pearl Tower and various skyscrapers. The text is overlaid on this background.

[Event Information](#) [Speakers](#) [Call for Proposal](#) [Agenda](#) [Partnership](#) [History](#) [Location](#) [Dashboard](#) [中文](#) [REGISTER](#)

## COSCon 2019 Coming!

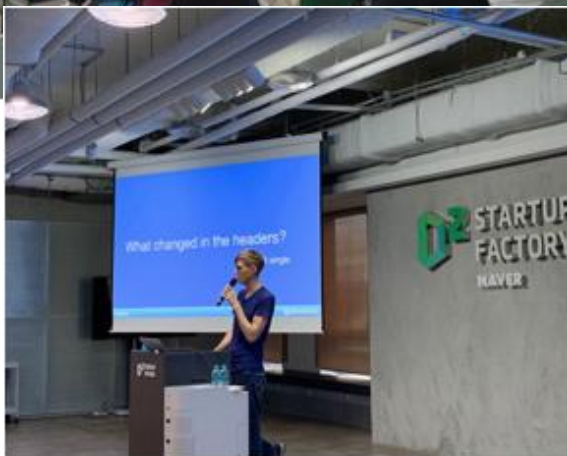
 Date 2019-11-02 09:00 ~ 11-03 17:00

 Location East China Normal University (Zhongshan North Road Campus), Putuo District, Shanghai, Putuo, Shanghai

The event is organized by KAIYUANSHE

[REGISTER](#)

# Open Source Community





# Open Source Community

---



# Open Source Community

---



**GOOD HOSTING!!!!**

# Open Source Community

**Adrian Cole** @adriancole · 7월 16일  
Subtle, but if you look carefully, Naver made @Pinpoint\_APM @ASFSkyWalking and @zipkinproject cookies. They could have done kool-aid, but...

Pinpoint  
The t  
Thank  
Spec  
@adri  
@Tom  
and c

**Chris K Wensel** @cwensel · 7월 13일  
Had some serious FOMO this week.

Pinpoint  
The tw  
Thanks  
Specia  
@adria  
@Tomr  
and of

**Wu Sheng 吴晨** @wusheng1108 · 7월 12일  
My daughter is the center of event. Haha  
Glad to meet old and new friends there. Thanks everyone.

**Pinpoint** @Pinpoint\_APM · 7월 12일  
The two days of "Pinpoint Open House" event has finished. Thanks to everyone who has joined us in the event.  
Special Thanks to  
@adriancole @autoletics @wusheng1108 @rakyll @jcchavez  
@TommyLudwig @dtornkaew, Chenguoxi, Liumingyi  
and of course, members of Team Pinpoint



Cool! Thanks for @Pinpoint\_APM team.



1 9

**tetrateio** @tetrateio · 7월 11일  
Our team member @wusheng1108 is in Seoul, Korea at @Pinpoint\_APM's #opensource conference. He'll be presenting on the core concepts of @ASFSkyWalking and sharing with the crowd the latest look of SkyWalking's UI today.



**Jaana B. Dogan** @rakyll · 7월 11일  
No one cares about metrics, traces, logs, etc. We care about signaling behaviors and reasoning about the signals. We need a model that scales to humans not to machines. — @autoletics



1 4 12

이 스레드 보기

**José Carlos Chávez** @jcchavez · 7월 11일  
I really love this #zipkin cookies.



1 3 11



# Open Source Community

**Jaana B. Dogan** @rakyll · 7월 12일  
I am at NAVER.



2   16

**Jaana B. Dogan** @rakyll


NAVER wins the tech loo of all times.

트윗 번역하기



오후 6:47 · 2019년 7월 12일, NAVER Green Factory · Twitter for iPhone

**Jaana B. Dogan** @rakyll · 7월 11일  
This is just reflex.



0:04 조회수 2,8천회

9   2   26

**Adrian Cole** @adriancole · 7월 12일  
a thread to save!

**Jaana B. Dogan** @rakyll · 7월 11일  
At NAVER OpenSource Seminar. Bob Ross programming jokes are a thing in Korea too.  
[이 스레드 보기](#)



## Scaling Distributed Tracing

Distributed Tracing Workshop — Seoul — July 2019

**Autoletics** [Follow](#)  
jul 18 · 8 min read

Below I've listed the slides I presented on the first day of a distributed tracing workshop organized by Naver Corp. and held in Seoul, South Korea.

**Jaana B. Dogan** @rakyll · 7월 12일



3   42

**Elle Townsend** @\_elbtownsend · 7월 12일  
[@rakyll](#) 남에게 보내는 답글  
A dream of mine to visit NAVER

1

**schavez** · 7월 10일  
n but I think I got this one pretty clear.



2

# Open Source Community

---



# Open Source Community

---





# Open Source Community

---





# Microservice Challenges

---

## **More areas that can fail**

99% success rate over 10 components -> 90.5% success rate

Network error much more relevant

## **Increase in latency**

1s 99th percentile over 10 components -> approx. 80th percentile

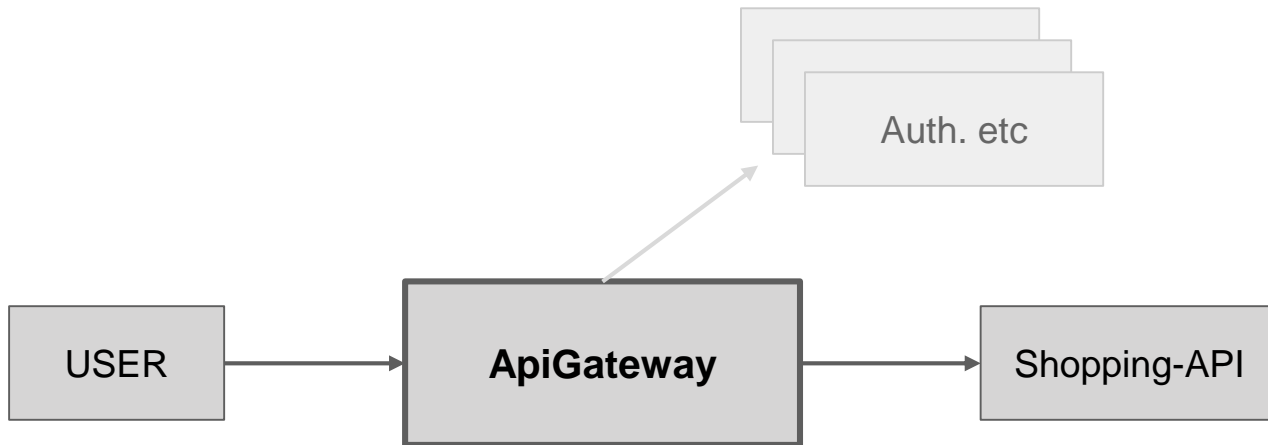
## **Observability greatly reduced**

Individual servers does not tell the whole story

Traditional way of troubleshooting no longer works

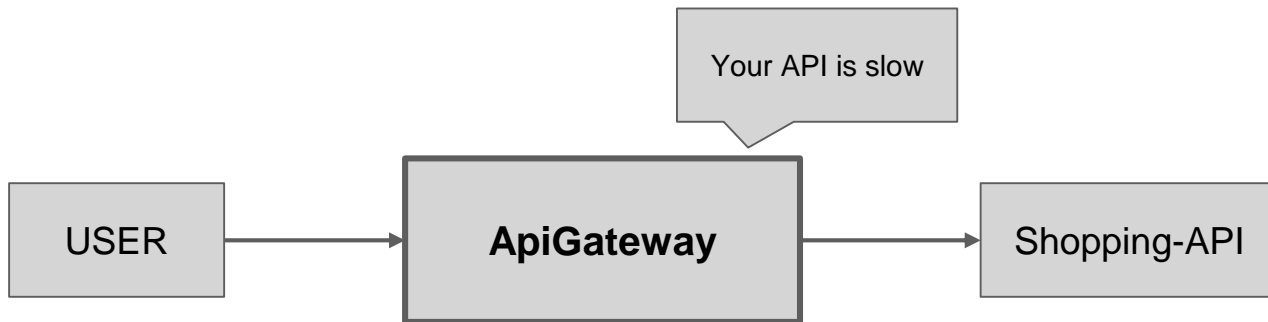
# Troubleshooting Microservices

---



# Troubleshooting Microservices

---

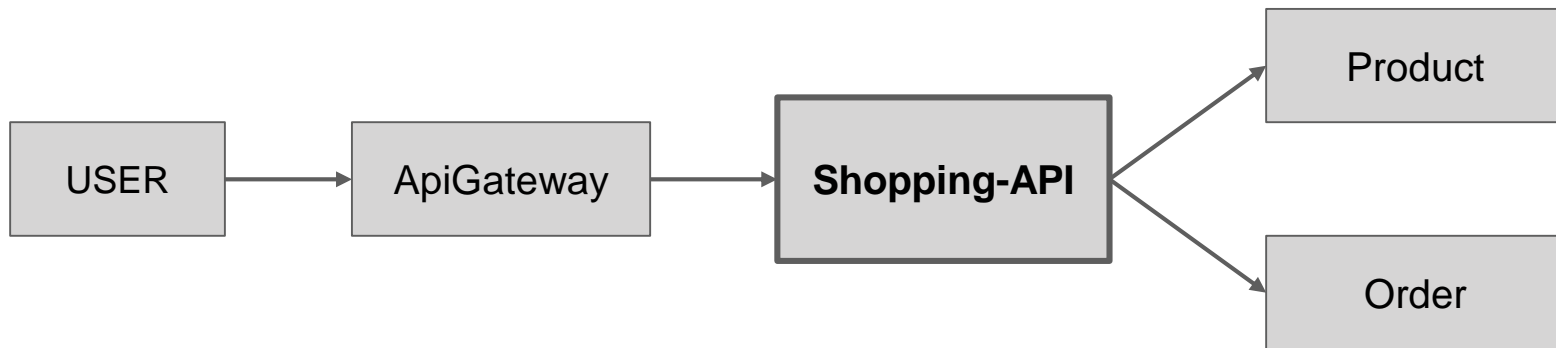


# Troubleshooting Microservices

```
[nio-8080-exec-9] o.s.web.client.RestTemplate : HTTP GET http://shopping.demo.pinpoint.com:8180/shopping/products
[nio-8080-exec-9] o.s.web.client.RestTemplate : Accept=[application/json, application/*+json]
[nio-8080-exec-9] o.s.web.client.RestTemplate : Response 200 OK
[nio-8080-exec-9] o.s.web.client.RestTemplate : Reading to [java.util.List<com.navercorp.pinpoint.demo.commons.vo.product.ProductInfo>]
[nio-8080-exec-9] c.n.p.d.a.service.ShoppingServiceImpl : GET http://shopping.demo.pinpoint.com:8180/shopping/products complete, took 2ms
[nio-8080-exec-8] o.s.w.s.m.m.a.HttpEntityMethodProcessor : Found 'Content-Type:application/json;charset=UTF-8' in response
[nio-8080-exec-8] o.s.w.s.m.m.a.HttpEntityMethodProcessor : Writing [(com.navercorp.pinpoint.demo.commons.vo.product.ProductInfo@8392e7f38, com.navercorp.pinpoint.demo.co (truncated)...
[nio-8080-exec-9] o.s.web.servlet.DispatcherServlet : Completed 200 OK
[nio-8080-exec-3] o.s.web.servlet.DispatcherServlet : GET "/v1/shopping/orders/30548a88-00a8-4f45-9f70-166c1afeb785", parameters={}
[nio-8080-exec-3] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped to public org.springframework.http.ResponseEntity<com.navercorp.pinpoint.demo.commons.vo.order.OrderInfo> com.navercorp.pinpoint
[nio-8080-exec-3] o.s.web.client.RestTemplate : HTTP GET http://shopping.demo.pinpoint.com:8180/shopping/orders/30548a88-00a8-4f45-9f70-166c1afeb785
[nio-8080-exec-3] o.s.web.client.RestTemplate : Accept=[application/json, application/*+json]
[nio-8080-exec-3] o.s.web.client.RestTemplate : Response 200 OK
[nio-8080-exec-3] o.s.web.client.RestTemplate : Reading to [com.navercorp.pinpoint.demo.commons.vo.order.OrderInfo]
[nio-8080-exec-3] c.n.p.d.a.service.ShoppingServiceImpl : GET http://shopping.demo.pinpoint.com:8180/shopping/orders/30548a88-00a8-4f45-9f70-166c1afeb785 complete, took 3ms
[nio-8080-exec-3] o.s.w.s.m.m.a.HttpEntityMethodProcessor : Found 'Content-Type:application/json;charset=UTF-8' in response
[nio-8080-exec-3] o.s.w.s.m.m.a.HttpEntityMethodProcessor : Writing [com.navercorp.pinpoint.demo.commons.vo.order.OrderInfo@4786d3eb]
[nio-8080-exec-3] o.s.web.servlet.DispatcherServlet : Completed 200 OK
[nio-8080-exec-10] o.s.web.servlet.DispatcherServlet : PATCH "/v1/shopping/orders/30548a88-00a8-4f45-9f70-166c1afeb785", parameters={}
[nio-8080-exec-10] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped to public org.springframework.http.ResponseEntity<java.lang.Boolean> com.navercorp.pinpoint.demo.apigw.controller.ApigwControlle
[nio-8080-exec-10] m.m.a.RequestResponseBodyMethodProcessor : Read 'application/json;charset=UTF-8' to [com.navercorp.pinpoint.demo.commons.vo.order.OrderPaymentParam@f3635b]
[nio-8080-exec-10] o.s.web.client.RestTemplate : HTTP PATCH http://shopping.demo.pinpoint.com:8180/shopping/orders/30548a88-00a8-4f45-9f70-166c1afeb785
[nio-8080-exec-10] o.s.web.client.RestTemplate : Accept=[application/json, application/*+json]
[nio-8080-exec-10] o.s.web.client.RestTemplate : Writing [com.navercorp.pinpoint.demo.commons.vo.order.OrderPaymentParam@f3635b] with org.springframework.http.converter.json.MappingJso
[nio-8080-exec-10] o.s.web.client.RestTemplate : Response 200 OK
[nio-8080-exec-1] o.s.web.client.RestTemplate : Reading to [java.lang.Boolean]
[nio-8080-exec-1] c.n.p.d.a.service.ShoppingServiceImpl : PATCH http://shopping.demo.pinpoint.com:8180/shopping/orders/f04d58a9-9b0a-4546-85b5-6f24a0937e2f complete, took 154ms
[nio-8080-exec-1] o.s.w.s.m.m.a.HttpEntityMethodProcessor : Found 'Content-Type:application/json;charset=UTF-8' in response
[nio-8080-exec-1] o.s.w.s.m.m.a.HttpEntityMethodProcessor : Writing [true]
[nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed 200 OK
[nio-8080-exec-8] o.s.web.servlet.DispatcherServlet : GET "/v1/shopping/products", parameters={}
[nio-8080-exec-8] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped to public org.springframework.http.ResponseEntity<java.util.List<com.navercorp.pinpoint.demo.commons.vo.product.ProductInfo>> co
[nio-8080-exec-8] o.s.web.client.RestTemplate : HTTP GET http://shopping.demo.pinpoint.com:8180/shopping/products
[nio-8080-exec-8] o.s.web.client.RestTemplate : Accept=[application/json, application/*+json]
[nio-8080-exec-8] o.s.web.client.RestTemplate : Writing [com.navercorp.pinpoint.demo.commons.vo.order.OrderPaymentParam@1c2b21c] with org.springframework.http.converter.js
[nio-8080-exec-8] o.s.web.client.RestTemplate : Response 200 OK
[nio-8080-exec-8] o.s.web.client.RestTemplate : Reading to [java.lang.Boolean]
[nio-8080-exec-4] o.s.web.client.RestTemplate : PATCH http://shopping.demo.pinpoint.com:8180/shopping/orders/f04d58a9-9b0a-4546-85b5-6f24a0937e2f complete, took 6340ms
[nio-8080-exec-4] o.s.web.client.RestTemplate : Found 'Content-Type:application/json;charset=UTF-8' in response
[nio-8080-exec-4] o.s.web.client.RestTemplate : Writing [true]
[nio-8080-exec-4] o.s.web.client.RestTemplate : Completed 200 OK
[nio-8080-exec-4] o.s.web.client.RestTemplate : Response 200 OK
[nio-8080-exec-4] o.s.web.client.RestTemplate : Reading to [java.lang.Boolean]
[nio-8080-exec-5] o.s.web.client.RestTemplate : PATCH http://shopping.demo.pinpoint.com:8180/shopping/orders/30548a88-00a8-4f45-9f70-166c1afeb785 complete, took 9334ms
[nio-8080-exec-5] o.s.web.client.RestTemplate : Found 'Content-Type:application/json;charset=UTF-8' in response
[nio-8080-exec-5] o.s.web.client.RestTemplate : Writing [true]
[nio-8080-exec-5] o.s.web.servlet.DispatcherServlet : Completed 200 OK
[nio-8080-exec-5] o.s.web.servlet.DispatcherServlet : Reading to [java.lang.Boolean]
[nio-8080-exec-10] c.n.p.d.a.service.ShoppingServiceImpl : PATCH http://shopping.demo.pinpoint.com:8180/shopping/orders/30548a88-00a8-4f45-9f70-166c1afeb785 complete, took 9334ms
[nio-8080-exec-10] o.s.w.s.m.m.a.HttpEntityMethodProcessor : Found 'Content-Type:application/json;charset=UTF-8' in response
[nio-8080-exec-10] o.s.w.s.m.m.a.HttpEntityMethodProcessor : Writing [true]
```

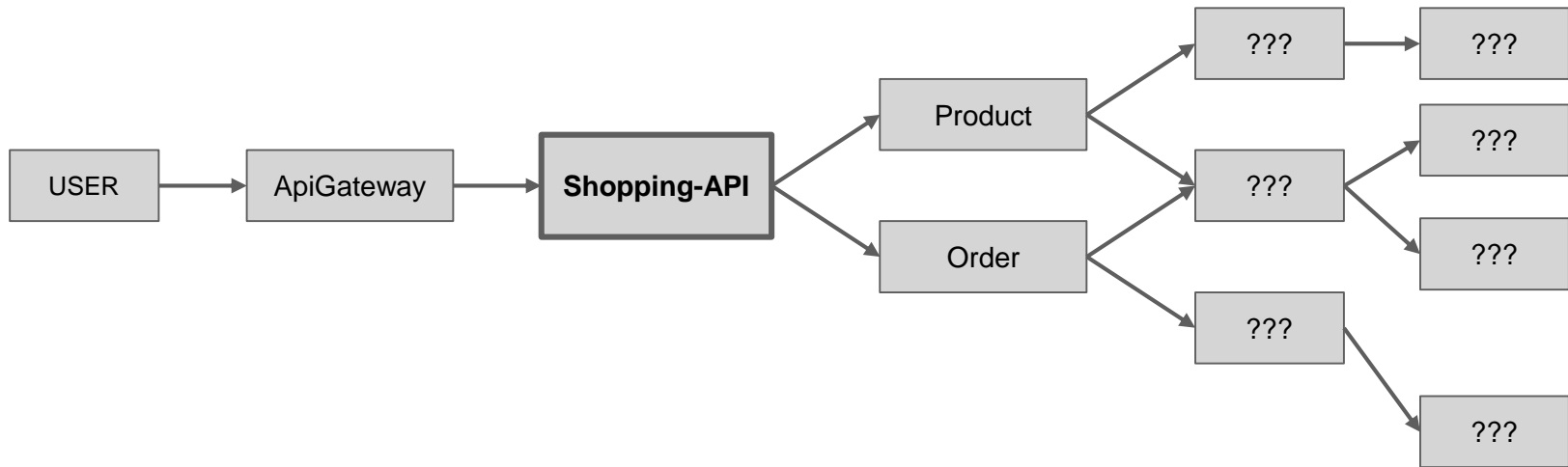
# Troubleshooting Microservices

---

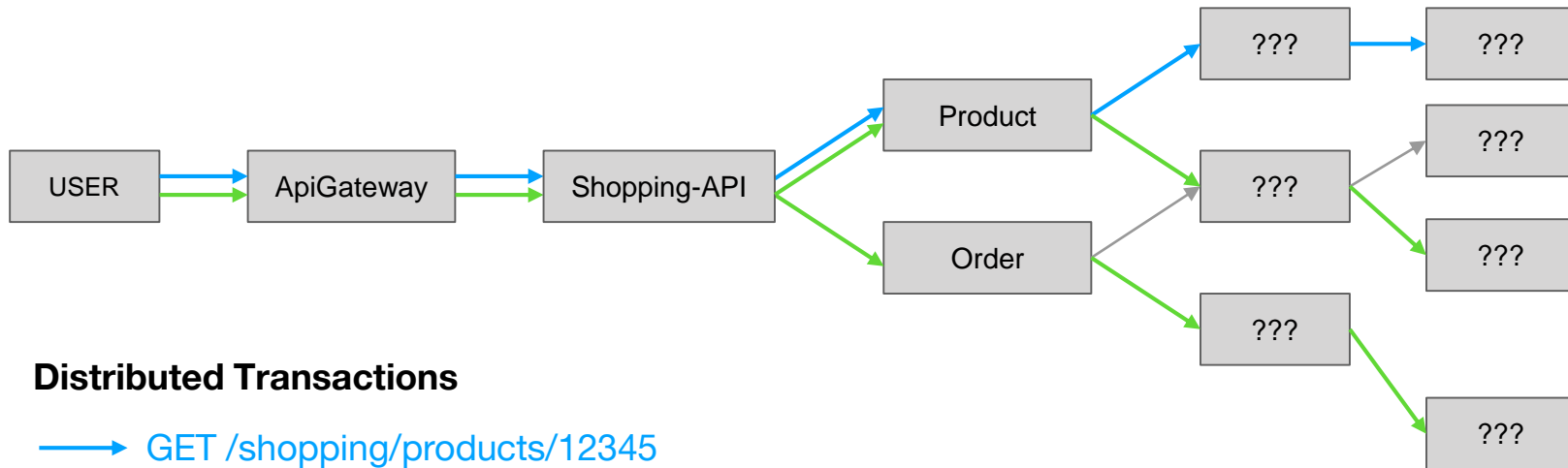


# Troubleshooting Microservices

---



# Troubleshooting Microservices

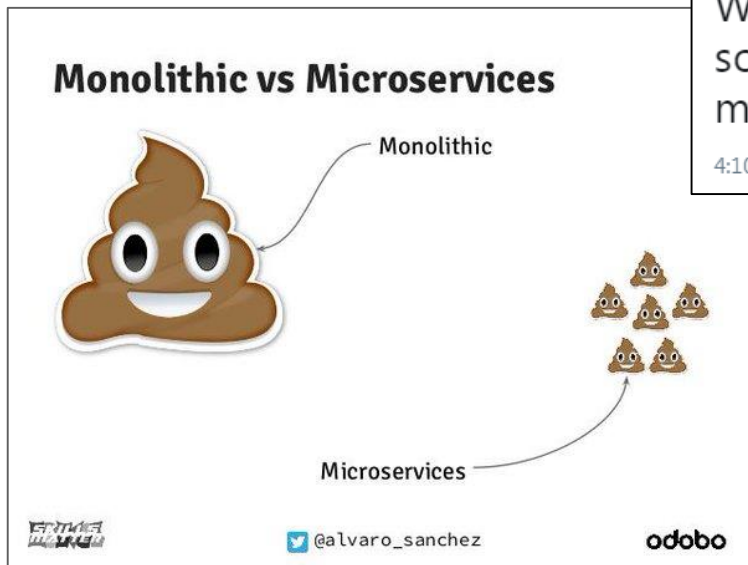


## Distributed Transactions

→ GET /shopping/products/12345

→ POST /shopping/orders

# Troubleshooting Microservices



@honest\_update 트위터



# So what do we need?

---

## Context

Identify as part of the same request within a single service instance

Identify as part of the same transaction within distributed service instances

Context propagation across thread, or process boundary

## Order

Timestamp not enough

Distributed Nodes - time skew

Asynchronous Processing - delayed execution

## Structure

Order - 1 dimensional

Call tree requires depth as well

# Pinpoint

---

## **Call Stack Trace**

Traces everything that happens in a single instance

Context propagated via thread-local

Order and structure inherently provided by emulating the call stack

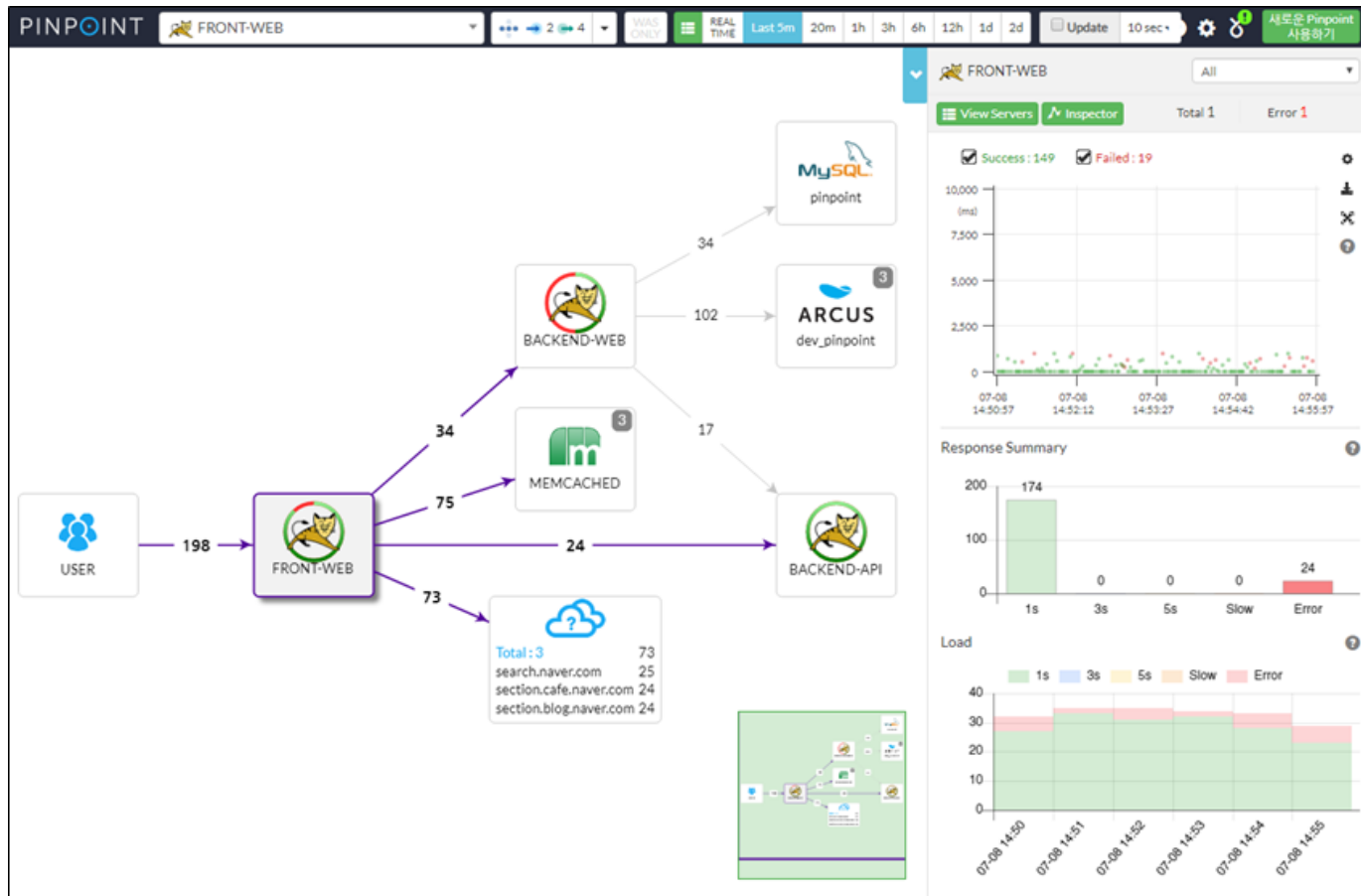
## **Distributed Transaction Trace**

Stitches multiple call stacks under the same transaction

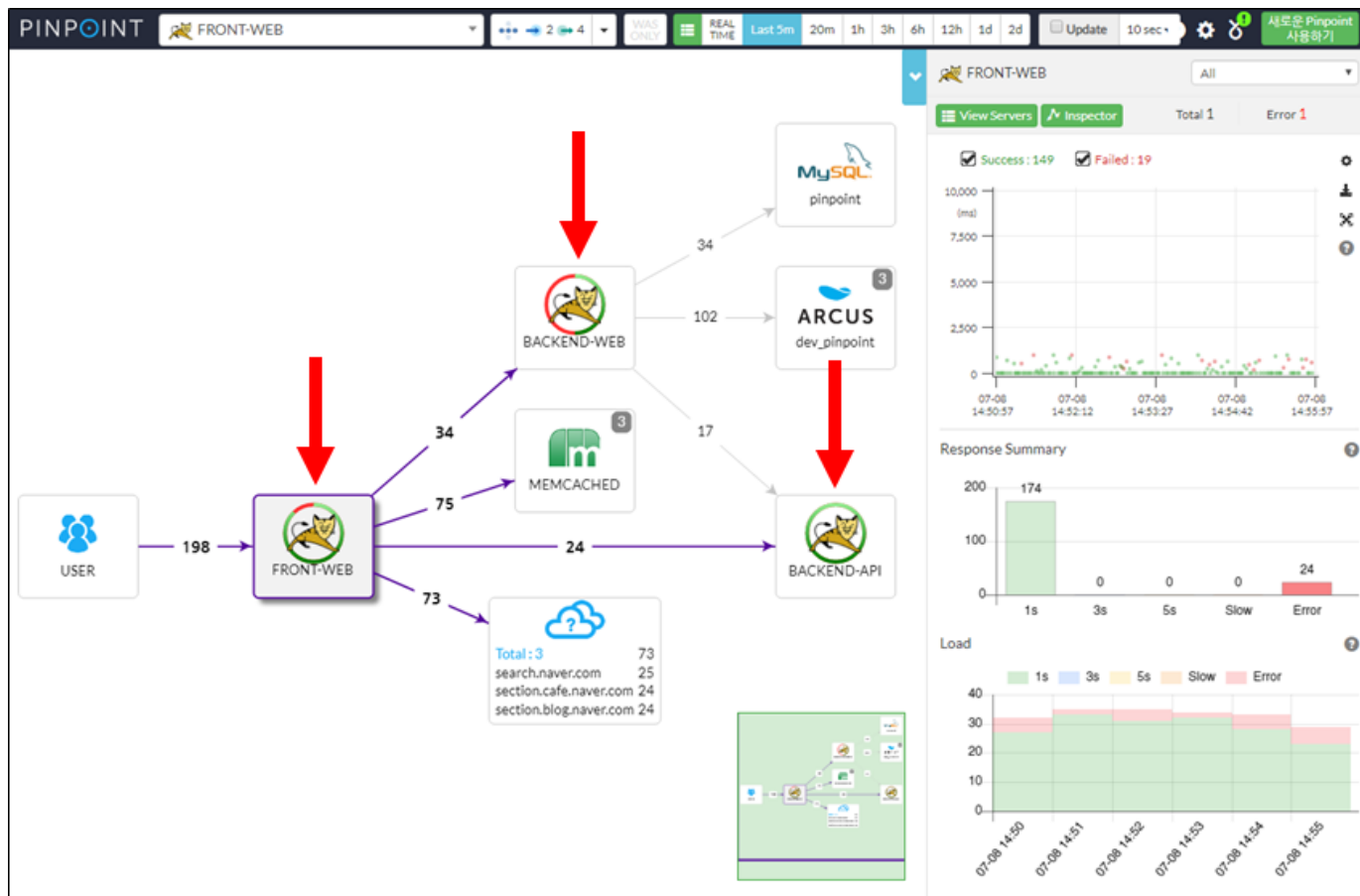
Context propagated via RPC

Order and structure via parent-child relationship of distributed call stacks

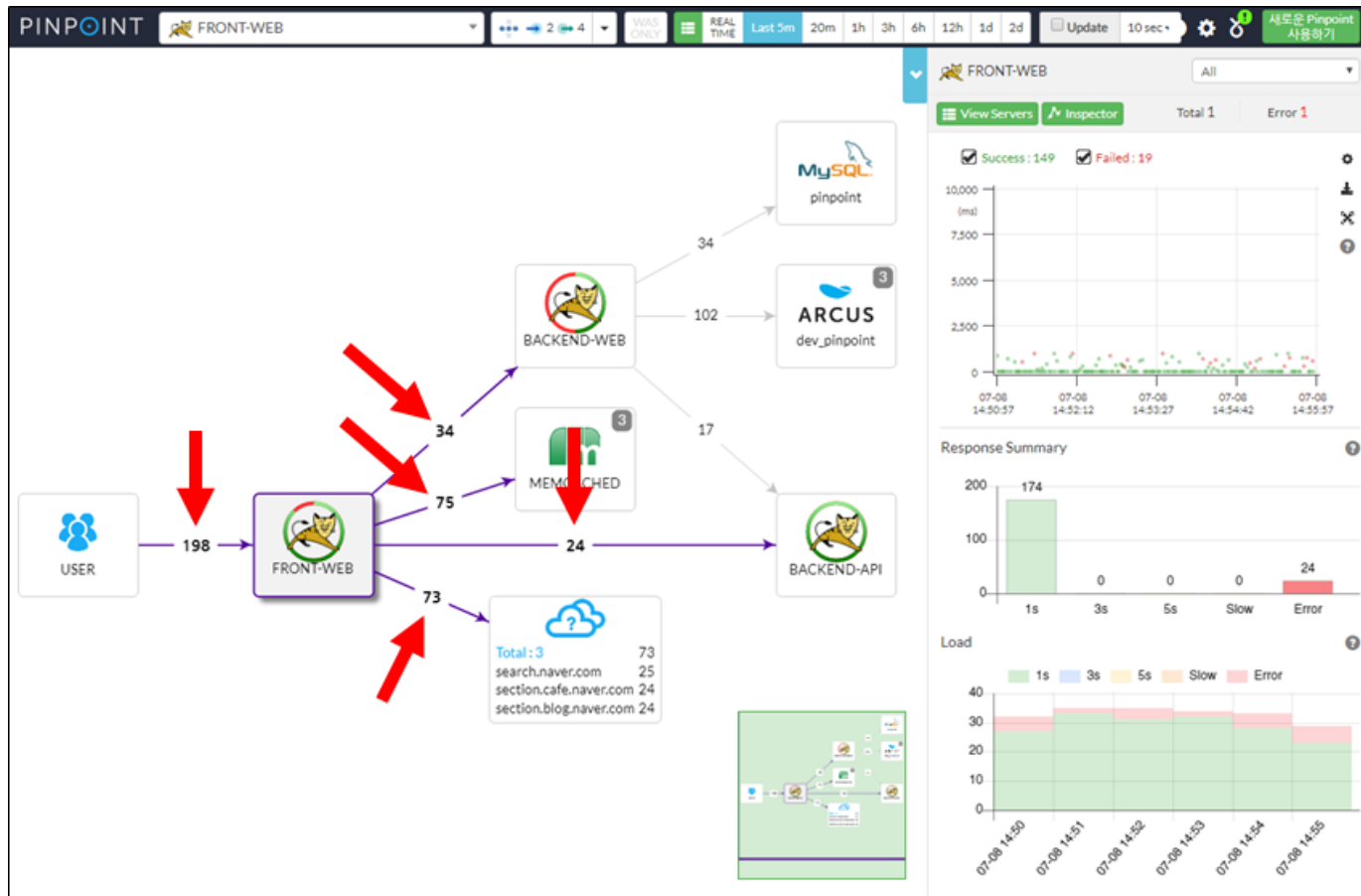
# Server Map



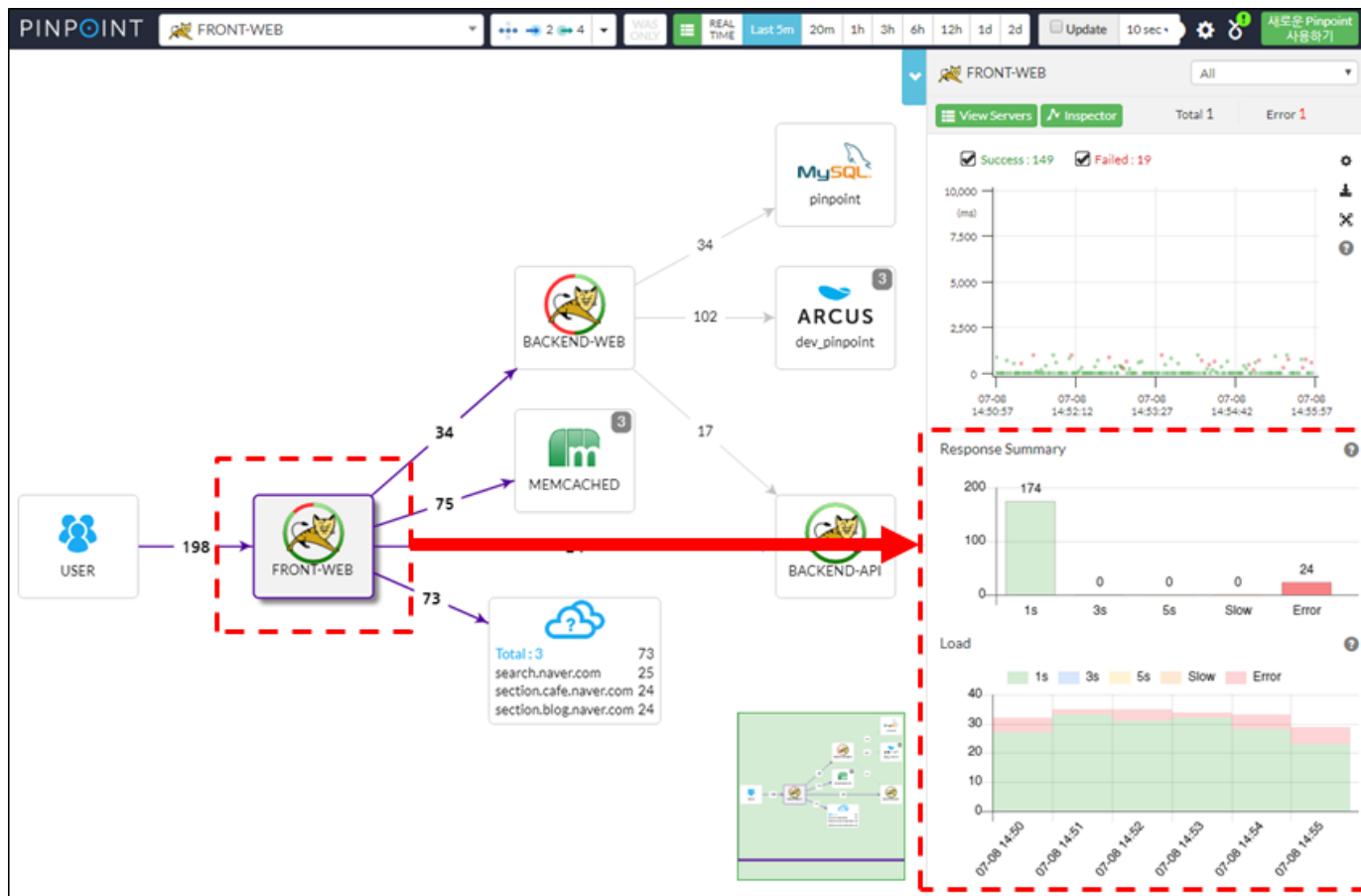
# Server Map



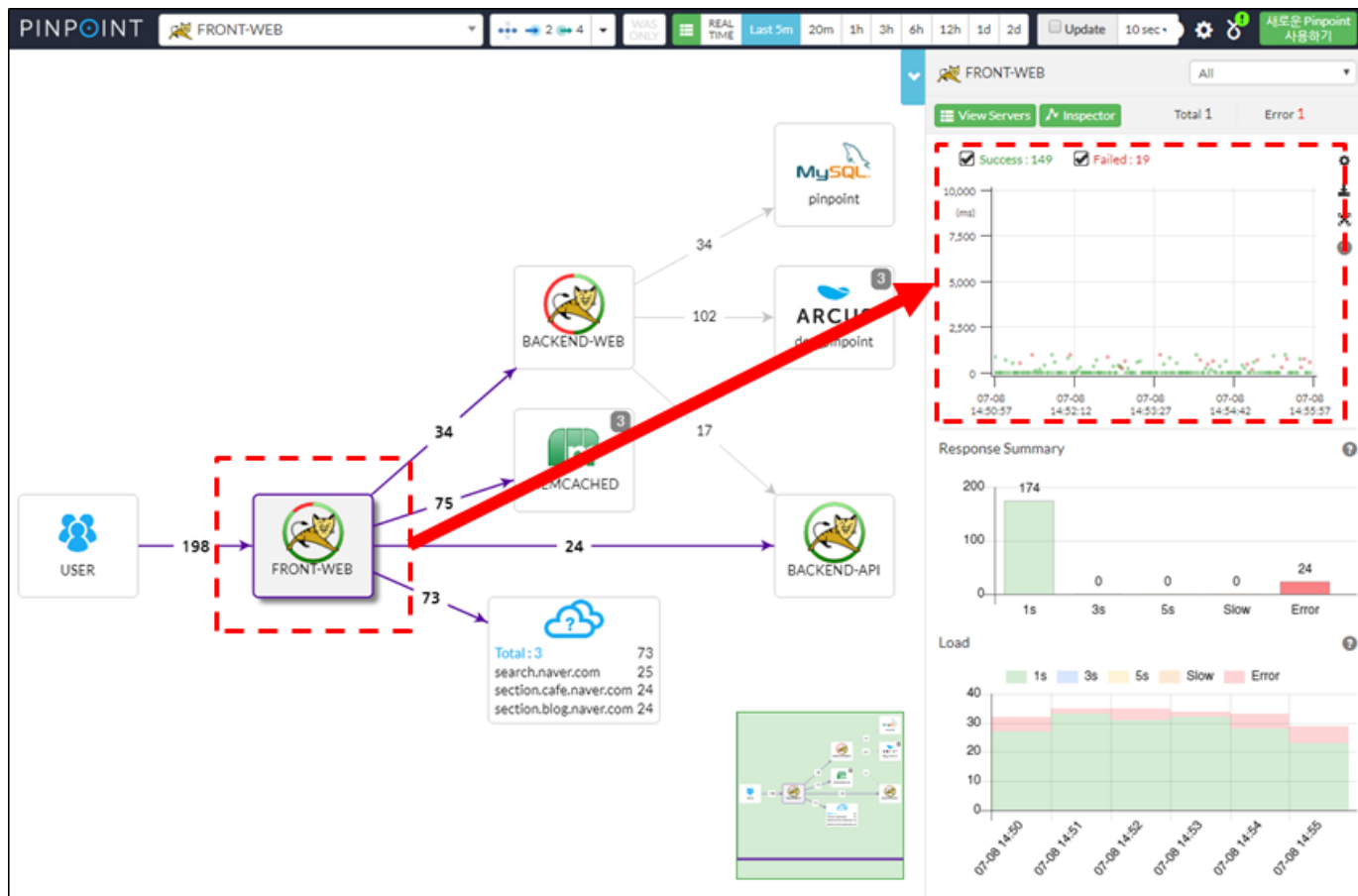
# Server Map



# Server Map



# Server Map



# Distributed Call Tree

Application : /emeroad.pinpoint      TransactionId : FrontWAS2^1563931395270^189...      AgentId : FrontWAS2      ApplicationName : FRONT-WEB

Call Tree    Server Map    Timeline    Mixed View    nelo    Self >=    1000(ms)    🔍 ↓    ?    Complete    ↻

Method	Argument	Start Time	Gap(ms)	Exec(ms)	Exec(%)	Self(ms)	Class	API	Application	Agent
Servlet Process	/emeroad.pinpoint	17:30:48 004	0	295	<div style="width: 100%;"></div>	0		TOMCAT	FRONT-WEB	FrontWAS2
⊕ http.status.code	200									
⊕ REMOTE_ADDRESS	127.0.0.1									
invoke(Request request, Response response)		17:30:48 004	0	295	<div style="width: 100%;"></div>	0	StandardHostValve	TOMCAT_ME...	FRONT-WEB	FrontWAS2
doGet(HttpServletRequest request, HttpServletResponse response)		17:30:48 004	0	295	<div style="width: 100%;"></div>	2	FrameworkServlet	SPRING	FRONT-WEB	FrontWAS2
demo2()		17:30:48 006	2	293	<div style="width: 100%;"></div>	280	DemoController	SPRING_BE...	FRONT-WEB	FrontWAS2
execute(HttpUriRequest request, Response response)		17:30:48 286	280	13	<div style="width: 100%;"></div>	0	CloseableHttpCli...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
open(HttpRoute route, HttpContext context)	dev-pinpoint-workload	17:30:48 286	0	0	<div style="width: 100%;"></div>	0	AbstractPooledCo...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
execute(HttpRequest request, HttpServletResponse response)	/backendweb.pinpoint	17:30:48 286	0	13	<div style="width: 100%;"></div>	13	HttpRequestExecu...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
⊕ http.status.code	200									
⊕ http.io	write: 0ms, read: 13m									
Servlet Process	/backendweb.pinpoint	17:30:48 286	0	12	<div style="width: 100%;"></div>	0		TOMCAT	BACKEND-WEB	BackendWAS1
⊕ http.status.code	200									
⊕ REMOTE_ADDRESS	██████████████████									
invoke(Request request, Response response)		17:30:48 286	0	12	<div style="width: 100%;"></div>	0	StandardHostValve	TOMCAT_ME...	BACKEND-WEB	BackendWAS1
doPost(HttpServletRequest request, HttpServletResponse response)		17:30:48 286	0	12	<div style="width: 100%;"></div>	9	FrameworkServlet	SPRING	BACKEND-WEB	BackendWAS1
backendweb()		17:30:48 295	9	3	<div style="width: 100%;"></div>	0	DemoController	SPRING_BE...	BACKEND-WEB	BackendWAS1
⊕ http.status.code	200									
⊕ REMOTE_ADDRESS	██████████████████									
CacheServiceTop1		17:30:48 295	0	1	<div style="width: 100%;"></div>	0	CacheServiceTop1	SPRING_BE...	BACKEND-WEB	BackendWAS1



# Distributed Call Tree

Application : /emeroad.pinpoint      TransactionId : FrontWAS2^1563931395270^189...      AgentId : FrontWAS2      ApplicationName : FRONT-WEB

Call Tree    Server Map    Timeline    Mixed View    nelo    Self >=    1000(ms)    🔍 ↓    ?    Complete    ↻

Method	Argument	Start Time	Gap(ms)	Exec(ms)	Exec(%)	Self(ms)	Class	API	Application	Agent
Servlet Process	/emeroad.pinpoint	17:30:48 004	0	295	<div style="width: 100%;"></div>	0		TOMCAT	FRONT-WEB	FrontWAS2
+ http.status.code	200									
+ REMOTE_ADDRESS	127.0.0.1									
invoke(Request request, Response response)		17:30:48 004	0	295	<div style="width: 100%;"></div>	0	StandardHostValve	TOMCAT_ME...	FRONT-WEB	FrontWAS2
doGet(HttpServletRequest request, HttpSe		17:30:48 004	0	295	<div style="width: 100%;"></div>	2	FrameworkServlet	SPRING	FRONT-WEB	FrontWAS2
demo2()		17:30:48 006	2	293	<div style="width: 100%;"></div>	280	DemoController	SPRING_BE...	FRONT-WEB	FrontWAS2
execute(HttpUriRequest request, Res		17:30:48 286	280	13	<div style="width: 5%;"></div>	0	CloseableHttpCli...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
open(HttpRoute route, HttpContext dev-pinpoint-workload		17:30:48 286	0	0	<div style="width: 0%;"></div>	0	AbstractPooledCo...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
execute(HttpServletRequest request, Http /backendweb.pinpoint		17:30:48 286	0	13	<div style="width: 5%;"></div>	13	HttpRequestExecu...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
+ http.status.code	200									
+ http.io	write: 0ms, read: 13m									
Servlet Process	/backendweb.pinpoint	17:30:48 286	0	12	<div style="width: 100%;"></div>	0		TOMCAT	BACKEND-WEB	BackendWAS1
+ http.status.code	200									
+ REMOTE_ADDRESS	██████████████████									
invoke(Request request, Resp		17:30:48 286	0	12	<div style="width: 100%;"></div>	0	StandardHostValve	TOMCAT_ME...	BACKEND-WEB	BackendWAS1
doPost(HttpServletRequest		17:30:48 286	0	12	<div style="width: 100%;"></div>	9	FrameworkServlet	SPRING	BACKEND-WEB	BackendWAS1
backendweb()		17:30:48 295	9	3	<div style="width: 25%;"></div>	0	DemoController	SPRING_BE...	BACKEND-WEB	BackendWAS1
+ http.io		17:30:48 295	0	1	<div style="width: 33%;"></div>	0	CacheServiceImp1	SPRING_BE...	BACKEND-WEB	BackendWAS1

# Distributed Call Tree

Application : /emeroad.pinpoint      TransactionId : FrontWAS2^1563931395270^189...      AgentId : FrontWAS2      ApplicationName : FRONT-WEB

Call Tree    Server Map    Timeline    Mixed View    nelo    Self >=    1000(ms)    🔍 ↓    ?    Complete    ↻

Method	Argument	Start Time	Gap(ms)	Exec(ms)	Exec(%)	Self(ms)	Class	API	Application	Agent
Servlet Process	/emeroad.pinpoint	17:30:48 004	0	295	<div style="width: 100%;"></div>	0		TOMCAT	FRONT-WEB	FrontWAS2
⊕ http.status.code	200									
⊕ REMOTE_ADDRESS	127.0.0.1									
invoke(Request request, Response response)		17:30:48 004	0	295	<div style="width: 100%;"></div>	0	StandardHostValve	TOMCAT_ME...	FRONT-WEB	FrontWAS2
doGet(HttpServletRequest request, HttpSe		17:30:48 004	0	295	<div style="width: 100%;"></div>	2	FrameworkServlet	SPRING	FRONT-WEB	FrontWAS2
demo2()		17:30:48 006	2	293	<div style="width: 100%;"></div>	280	DemoController	SPRING_BE...	FRONT-WEB	FrontWAS2
execute(HttpUriRequest request, Res		17:30:48 286	280	13	<div style="width: 5%;"></div>	0	CloseableHttpCli...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
open(HttpRoute route, HttpContext	dev.pinpoint workload	17:30:48 286	0	0	<div style="width: 0%;"></div>	0	AbstractPooledCo...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
⊕ execute() <b>HttpClient.execute()</b>	point	17:30:48 286	0	13	<div style="width: 5%;"></div>	13	HttpRequestExecu...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
⊕ http.status.code	200									
⊕ http.io	write: 0ms, read: 13m									
Servlet Process	/backendweb.pinpoint	17:30:48 286	0	12	<div style="width: 100%;"></div>	0		TOMCAT	BACKEND-WEB	BackendWAS1
⊕ http.status.code	200									
⊕ REMOTE_ADDRESS	████████████████									
invoke(Request request, Resp		17:30:48 286	0	12	<div style="width: 100%;"></div>	0	StandardHostValve	TOMCAT_ME...	BACKEND-WEB	BackendWAS1
doPost(HttpServletRequest		17:30:48 286	0	12	<div style="width: 100%;"></div>	9	FrameworkServlet	SPRING	BACKEND-WEB	BackendWAS1
backendweb()		17:30:48 295	9	3	<div style="width: 25%;"></div>	0	DemoController	SPRING_BE...	BACKEND-WEB	BackendWAS1
⊕ execute()		17:30:48 295	0	1	<div style="width: 33%;"></div>	0	CacheServiceImp	SPRING_BE...	BACKEND-WEB	BackendWAS1

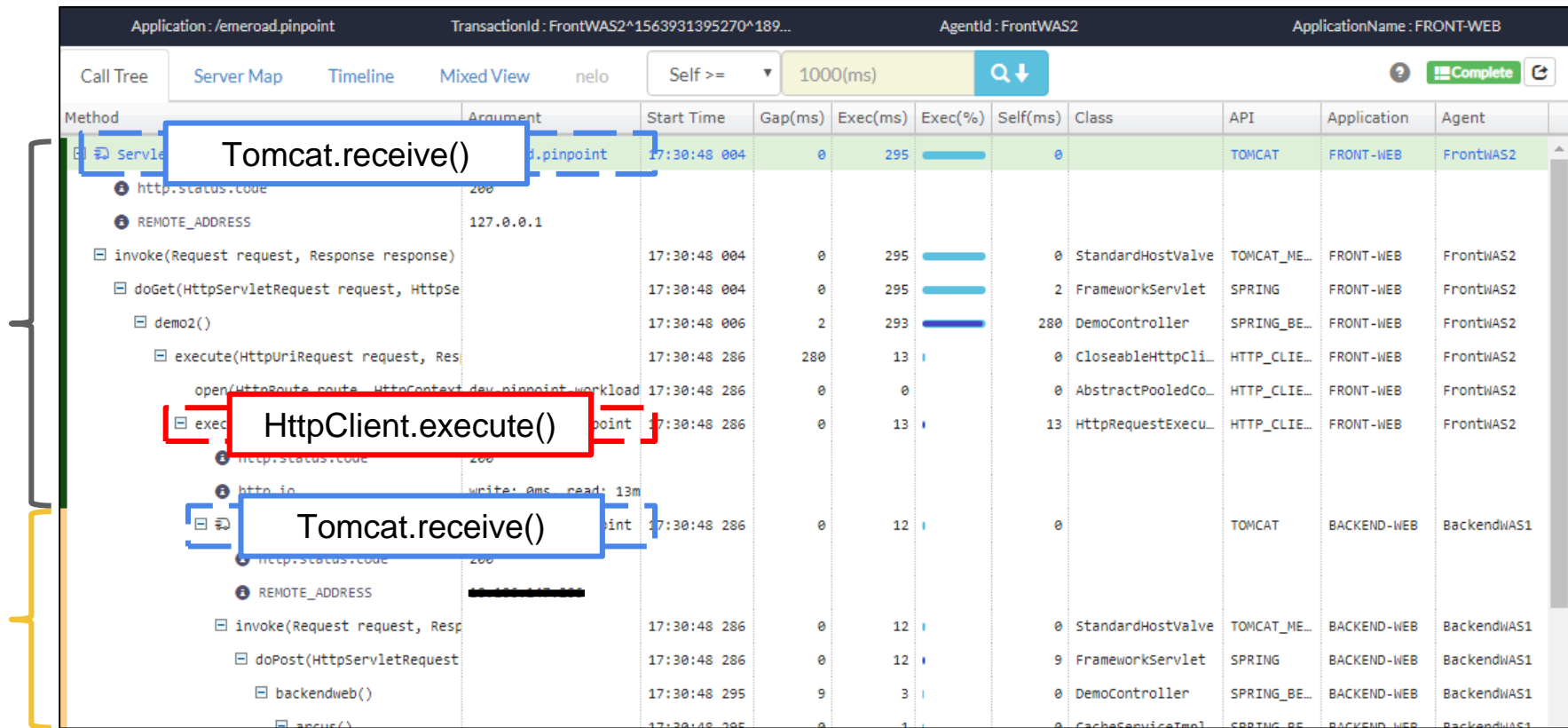
# Distributed Call Tree

Application : /emeroad.pinpoint      TransactionId : FrontWAS2^1563931395270^189...      AgentId : FrontWAS2      ApplicationName : FRONT-WEB

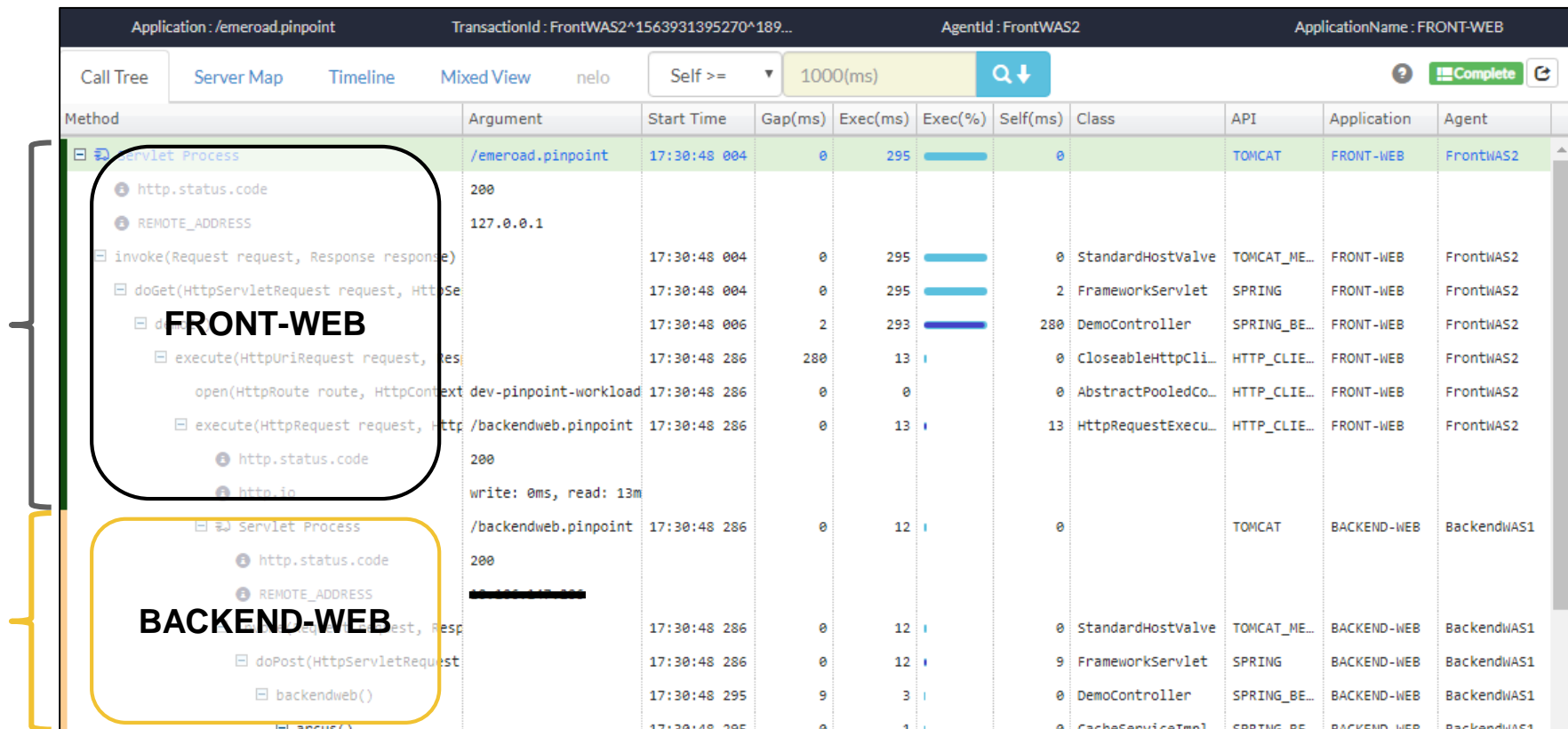
Call Tree    Server Map    Timeline    Mixed View    nelo    Self >=    1000(ms)    🔍 ↓    ?    Complete    ↻

Method	Argument	Start Time	Gap(ms)	Exec(ms)	Exec(%)	Self(ms)	Class	API	Application	Agent
Servlet Process	/emeroad.pinpoint	17:30:48 004	0	295		0		TOMCAT	FRONT-WEB	FrontWAS2
http.status.code	200									
REMOTE_ADDRESS	127.0.0.1									
invoke(Request request, Response response)		17:30:48 004	0	295		0	StandardHostValve	TOMCAT_ME...	FRONT-WEB	FrontWAS2
doGet(HttpServletRequest request, HttpSe		17:30:48 004	0	295		2	FrameworkServlet	SPRING	FRONT-WEB	FrontWAS2
demo2()		17:30:48 006	2	293		280	DemoController	SPRING_BE...	FRONT-WEB	FrontWAS2
execute(HttpUriRequest request, Res		17:30:48 286	280	13		0	CloseableHttpCli...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
open(HttpRoute route, HttpContext		17:30:48 286	0	0		0	AbstractPooledCo...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
execute() <b>HttpClient.execute()</b>		17:30:48 286	0	13		13	HttpRequestExecu...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
http.status.code	200									
http.io	write: 0ms, read: 13m									
Servlet Process	/backendweb.pinpoint	17:30:48 286	0	12		0		TOMCAT	BACKEND-WEB	BackendWAS1
http.status.code	200									
REMOTE_ADDRESS	████████████████									
invoke(Request request, Resp		17:30:48 286	0	12		0	StandardHostValve	TOMCAT_ME...	BACKEND-WEB	BackendWAS1
doPost(HttpServletRequest		17:30:48 286	0	12		9	FrameworkServlet	SPRING	BACKEND-WEB	BackendWAS1
backendweb()		17:30:48 295	9	3		0	DemoController	SPRING_BE...	BACKEND-WEB	BackendWAS1
cacheService()		17:30:48 295	0	1		0	CacheServiceImp	SPRING_BE...	BACKEND-WEB	BackendWAS1

# Distributed Call Tree



# Distributed Call Tree



# How it all works

---

**Call Stack Trace**

**Distributed Transaction Trace**

# Call Stack Trace

Application : /emeroad.pinpoint      TransactionId : FrontWAS2^1563931395270^189...      AgentId : FrontWAS2      ApplicationName : FRONT-WEB

Call Tree    Server Map    Timeline    Mixed View    nelo    Self >=    1000(ms)    🔍 ↓    ?    Complete    ↻

Method	Argument	Start Time	Gap(ms)	Exec(ms)	Exec(%)	Self(ms)	Class	API	Application	Agent
Servlet Process	/emeroad.pinpoint	17:30:48 004	0	295	<div style="width: 100%;"></div>	0		TOMCAT	FRONT-WEB	FrontWAS2
⊕ http.status.code	200									
⊕ REMOTE_ADDRESS	127.0.0.1									
⊖ invoke(Request request, Response response)		17:30:48 004	0	295	<div style="width: 100%;"></div>	0	StandardHostValve	TOMCAT_ME...	FRONT-WEB	FrontWAS2
⊖ doGet(HttpServletRequest request, HttpSe		17:30:48 004	0	295	<div style="width: 100%;"></div>	2	FrameworkServlet	SPRING	FRONT-WEB	FrontWAS2
⊖ demo2()		17:30:48 006	2	293	<div style="width: 100%;"></div>	280	DemoController	SPRING_BE...	FRONT-WEB	FrontWAS2
⊖ execute(HttpUriRequest request, Res		17:30:48 286	280	13	<div style="width: 5%;"></div>	0	CloseableHttpCli...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
open(HttpRoute route, HttpContext	dev-pinpoint-workload	17:30:48 286	0	0	<div style="width: 0%;"></div>	0	AbstractPooledCo...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
⊖ execute(HttpRequest request, Http	/backendweb.pinpoint	17:30:48 286	0	13	<div style="width: 5%;"></div>	13	HttpRequestExecu...	HTTP_CLIE...	FRONT-WEB	FrontWAS2
⊕ http.status.code	200									
⊕ http.io	write: 0ms, read: 13m									

`invoke(); -> doGet(); -> demo2();`

# Call Stack Trace

---

```
invoke(); -> doGet(); -> demo2();
```



```
invoke() {  
    doGet();  
}  
doGet() {  
    demo2();  
}  
demo2() {  
}
```



# Call Stack Trace

---

```
invoke(); -> doGet(); -> demo2();
```



```
invoke() {          doGet() {          demo2() {  
  doGet();          demo2();          }  
}                   }  
}
```



```
invoke() {  
  doGet() {  
    demo2() {  
    }  
  }  
}
```

# Call Stack Trace

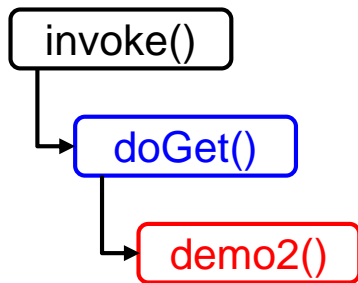
```
invoke(); -> doGet(); -> demo2();
```



```
invoke() {           doGet() {           demo2() {  
  doGet();           demo2();           }  
}                   }  
}
```



```
invoke() {  
  doGet() {  
    demo2() {  
    }  
  }  
}
```



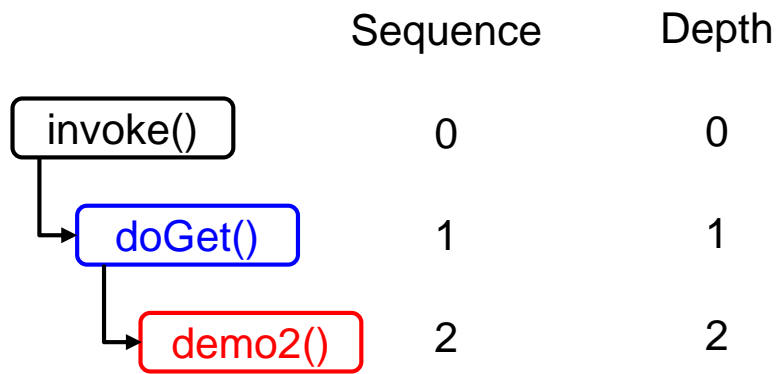
# Call Stack Trace

---

	Sequence	Depth
invoke()	0	
doGet()	1	
demo2()	2	

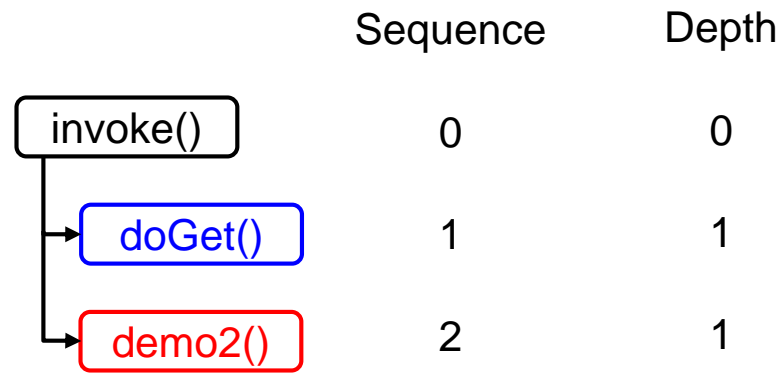
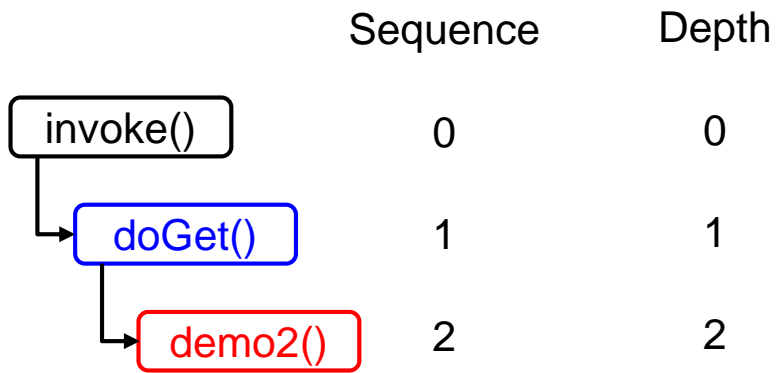
# Call Stack Trace

---



# Call Stack Trace

---



# Call Stack Trace

---

```
foo() {  
    fooInterceptor.before();  
    ...  
    fooInterceptor.after();  
}
```

# Call Stack Trace

---

```
invoke() {  
    invokeInterceptor.before();  
    doGet() {  
        doGetInterceptor.before();  
        demo2() {  
            demo2Interceptor.before();  
            demo2Interceptor.after();  
        }  
        doGetInterceptor.after();  
    }  
    invokeInterceptor.after();  
}
```

# Call Stack Trace

```
invoke() {  
    invokeInterceptor.before();  
    doGet() {  
        doGetInterceptor.before();  
        demo2() {  
            demo2Interceptor.before();  
            demo2Interceptor.after();  
        }  
        doGetInterceptor.after();  
    }  
    invokeInterceptor.after();  
}
```

**before()**

Increment sequence

Push on to the call stack

Seq.	Event	Depth
		3
		2
		1
0	invoke()	0

push



# Call Stack Trace

```
invoke() {  
    invokeInterceptor.before();  
    doGet() {  
        doGetInterceptor.before();  
        demo2() {  
            demo2Interceptor.before();  
            demo2Interceptor.after();  
        }  
        doGetInterceptor.after();  
    }  
    invokeInterceptor.after();  
}
```

## before()

Increment sequence

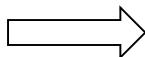
Push on to the call stack

Seq.	Event	Depth
		3
		2
1	doGet()	1
0	invoke()	0

push

# Call Stack Trace

```
invoke() {  
    invokeInterceptor.before();  
    doGet() {  
        doGetInterceptor.before();  
        demo2() {  
            demo2Interceptor.before();  
            demo2Interceptor.after();  
        }  
        doGetInterceptor.after();  
    }  
    invokeInterceptor.after();  
}
```



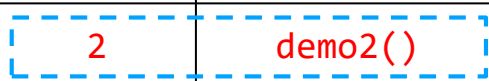
## before()

Increment sequence

Push on to the call stack

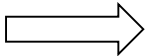
Seq.	Event	Depth
		3
2	demo2()	2
1	doGet()	1
0	invoke()	0

push



# Call Stack Trace

```
invoke() {  
  invokeInterceptor.before();  
  doGet() {  
    doGetInterceptor.before();  
    demo2() {  
      demo2Interceptor.before();  
      demo2Interceptor.after();  
    }  
    doGetInterceptor.after();  
  }  
  invokeInterceptor.after();  
}
```

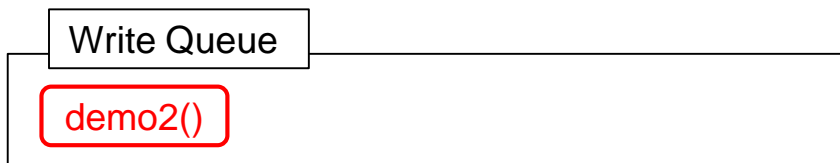


**after()**

Pop off the call stack  
Buffer to write queue

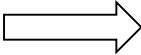
Seq.	Event	Depth
		3
		2
1	doGet()	1
0	invoke()	0

pop



# Call Stack Trace

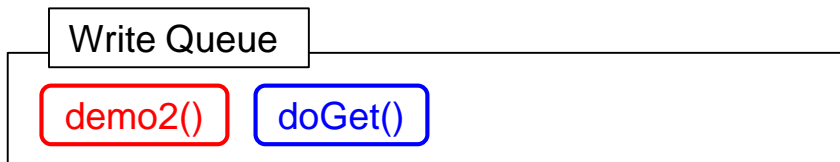
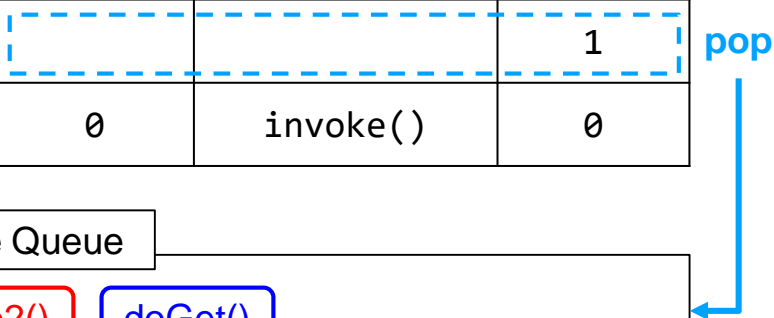
```
invoke() {  
  invokeInterceptor.before();  
  doGet() {  
    doGetInterceptor.before();  
    demo2() {  
      demo2Interceptor.before();  
      demo2Interceptor.after();  
    }  
    doGetInterceptor.after();  
  }  
  invokeInterceptor.after();  
}
```



**after()**

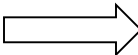
Pop off the call stack  
Buffer to write queue

Seq.	Event	Depth
		3
		2
		1
0	invoke()	0



# Call Stack Trace

```
invoke() {  
  invokeInterceptor.before();  
  doGet() {  
    doGetInterceptor.before();  
    demo2() {  
      demo2Interceptor.before();  
      demo2Interceptor.after();  
    }  
    doGetInterceptor.after();  
  }  
  invokeInterceptor.after();  
}
```

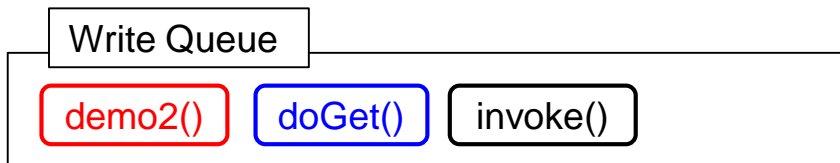


**after()**

Pop off the call stack  
Buffer to write queue

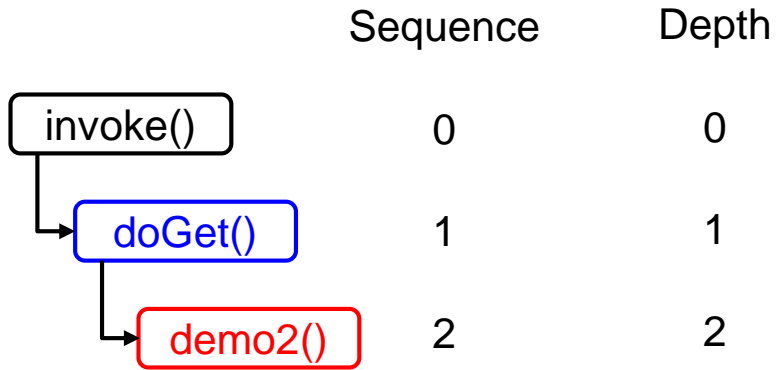
Seq.	Event	Depth
		3
		2
		1
		0

pop



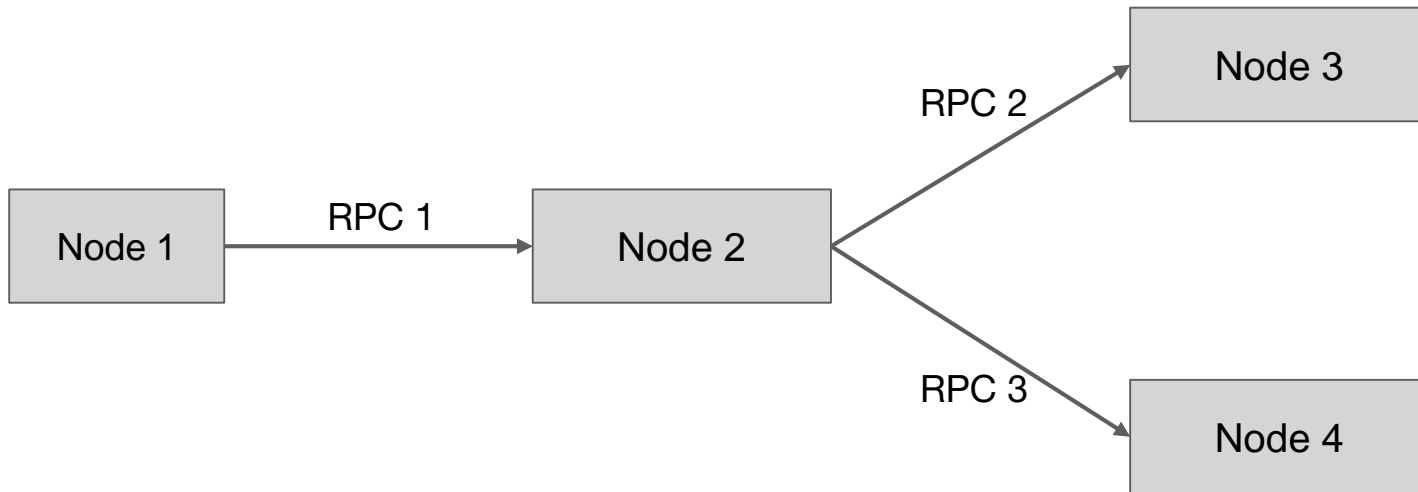
# Call Stack Trace

---



# Distributed Transaction Trace

---

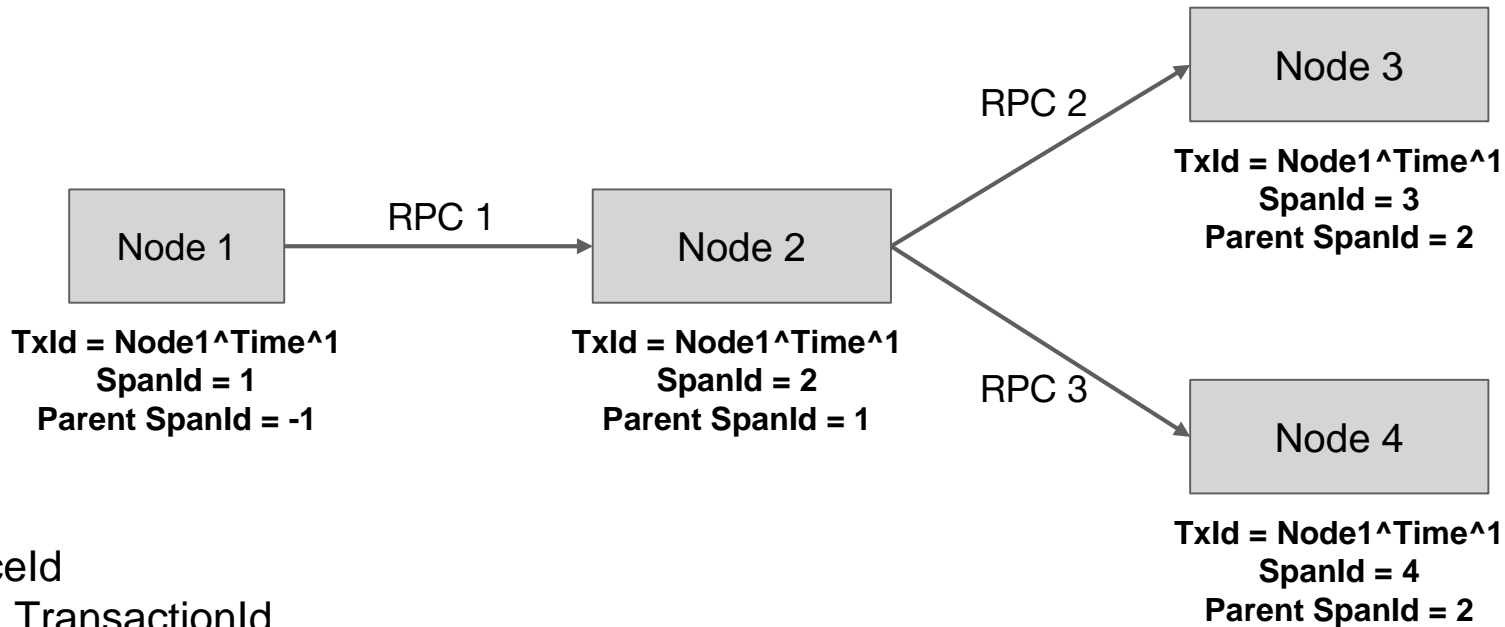


Find out the relationship between nodes connected by RPCs for a given transaction

Trace tag added to each requests

- HTTP : HttpHeaders

# Distributed Transaction Trace

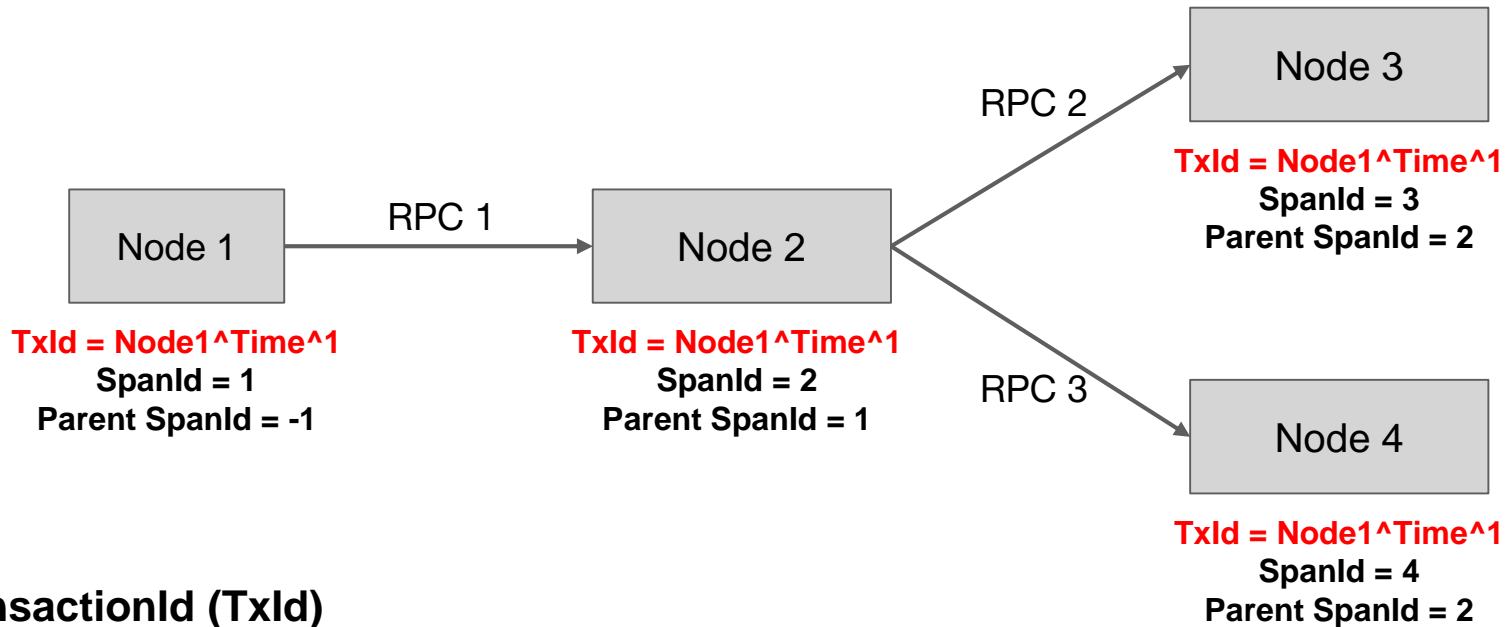


## TraceId

- TransactionId
- SpanId
- Parent SpanId



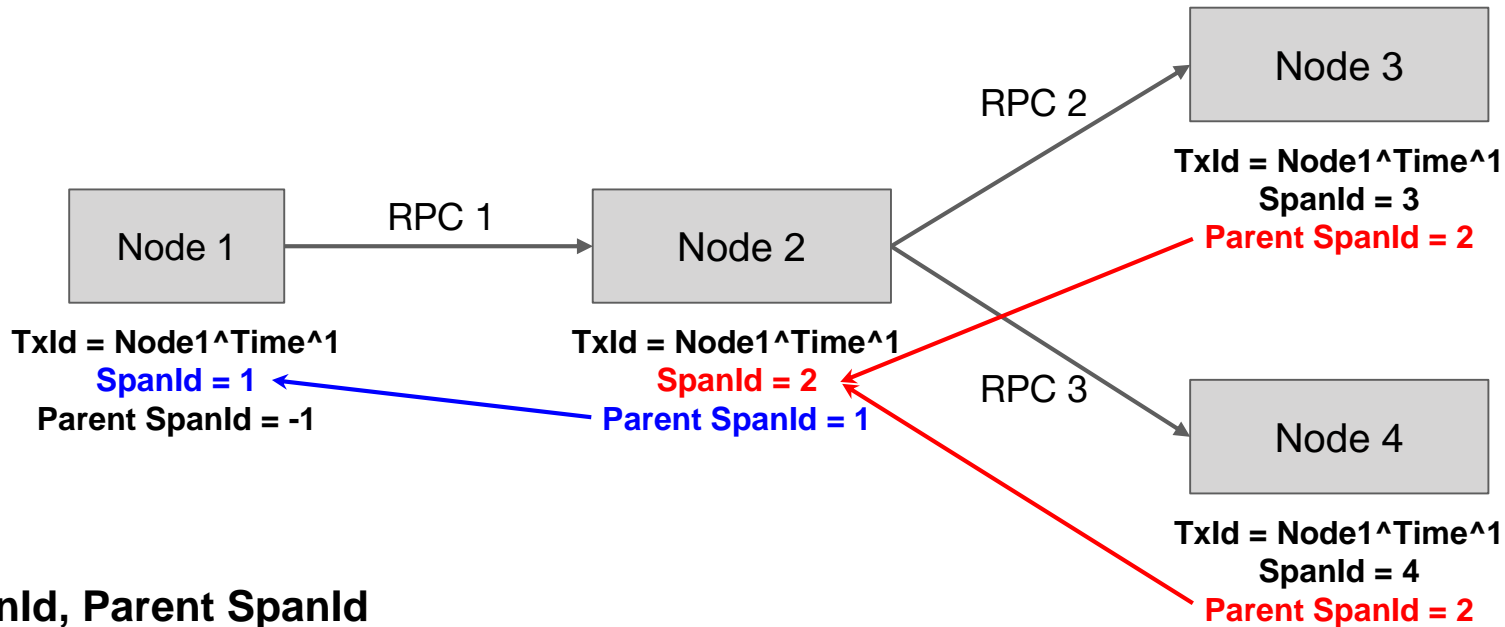
# Distributed Transaction Trace



## TransactionId (TxId)

Globally unique ID for a single transaction

# Distributed Transaction Trace



## SpanId, Parent SpanId

Id used to encode parent-child relationship between nodes

# Distributed Call Tree

Application : /emeroad.pinpoint      TransactionId : FrontWAS2^1563931395270^189...      AgentId : FrontWAS2      ApplicationName : FRONT-WEB

Call Tree    Server Map    Timeline    Mixed View    nelo    Self >=    1000(ms)    🔍 ↓    Complete    ↻

Method	Argument	Start Time	Gap(ms)	Exec(ms)	Exec(%)	Self(ms)	Class	API	Application	Agent
Servlet Process	/emeroad.pinpoint	17:30:48 004	0	295		0		TOMCAT	FRONT-WEB	FrontWAS2
+ http.status.code	200									
+ REMOTE_ADDRESS	127.0.0.1									
invoke(Request request, Response response)		17:30:48 004								
doGet(HttpServletRequest request, HttpServletResponse response)		17:30:48 004								
demo2()		17:30:48 006								
execute(HttpUriRequest request, Response response)		17:30:48 286								
open(HttpRoute route, HttpContext context)	dev-pinpoint-workload	17:30:48 286								
execute(HttpRequest request, HttpServletResponse response)	/backendweb.pinpoint	17:30:48 286								
+ http.status.code	200									
+ http.io	write: 0ms, read: 13m									
Servlet Process	/backendweb.pinpoint	17:30:48 286								
+ http.status.code	200									
+ REMOTE_ADDRESS	████████████████████									
invoke(Request request, Response response)		17:30:48 286								
doPost(HttpServletRequest request, HttpServletResponse response)		17:30:48 286								
backendweb()		17:30:48 295								
arcus()		17:30:48 295								

PINPOINT FRONT-WEB    View Servers    Inspector    Total 1    Error 1

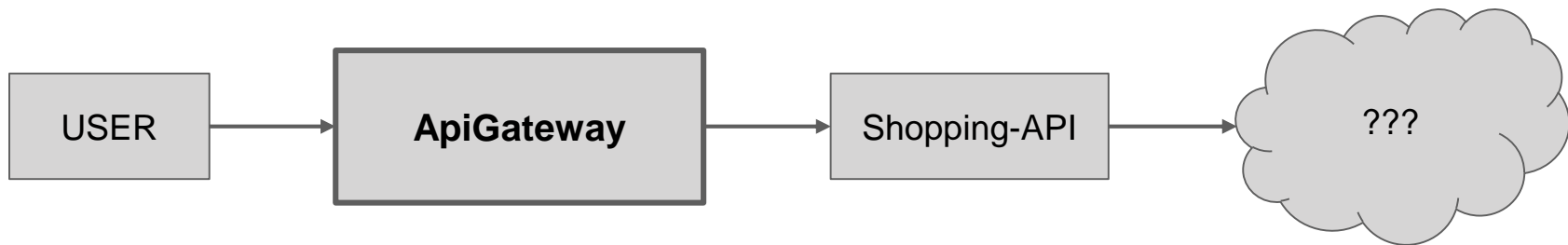
Success: 149    Failed: 19

Response Summary

Load

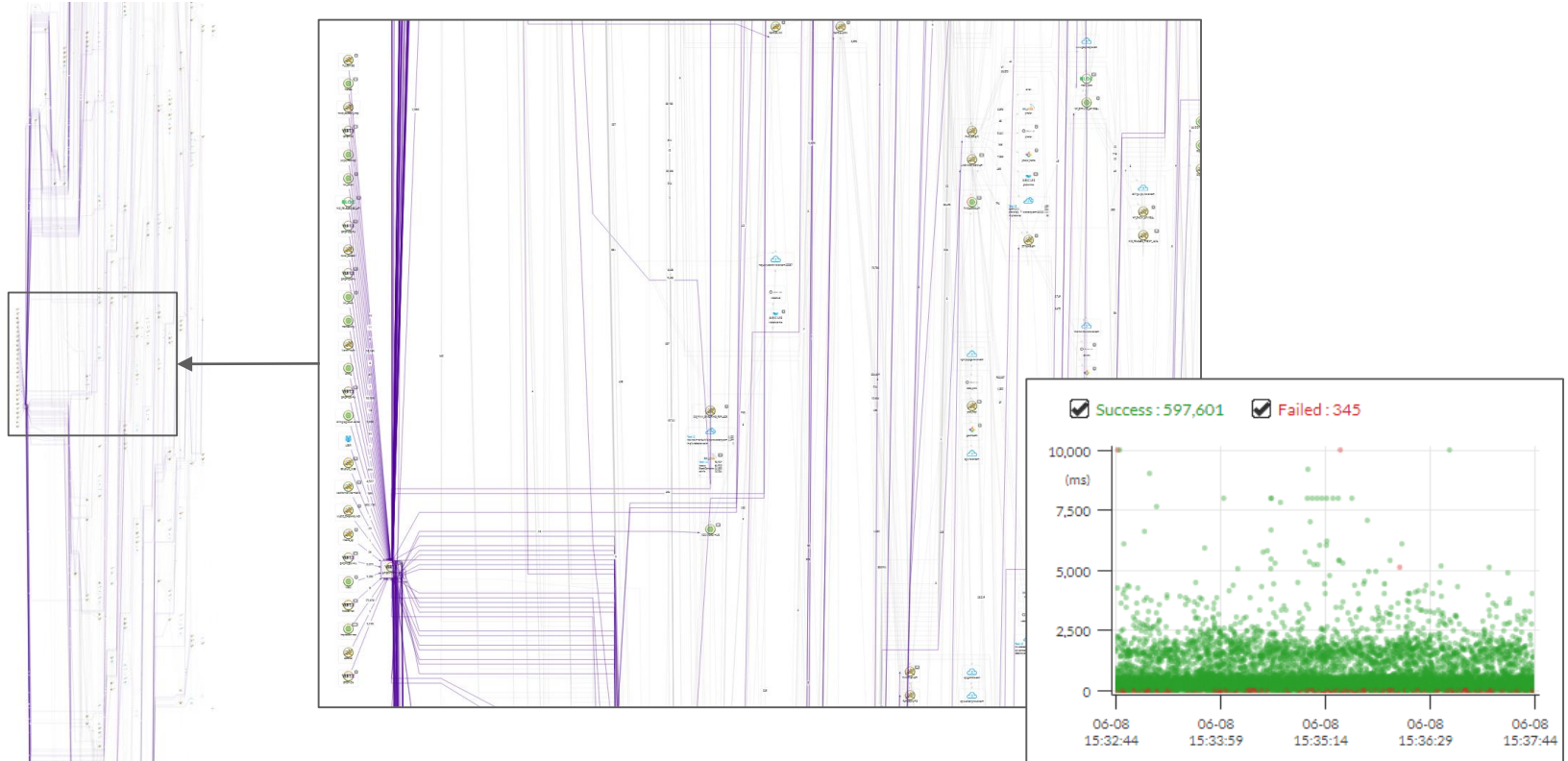
# Troubleshooting with Pinpoint

---



Let's troubleshoot our system using [Pinpoint](#)  
(There's a link @ [www.github.com/naver/pinpoint](http://www.github.com/naver/pinpoint))

# Troubleshooting with Pinpoint



---

# Future plans

# Future Plans

---



GOLANG



# Future Plans

---



**kubernetes**



**Istio** X





Thank you

---

# Q & A

**Homepage** : <https://naver.github.io/pinpoint>

**Github** : <https://github.com/naver/pinpoint>

**Twitter** : [https://twitter.com/Pinpoint\\_APM](https://twitter.com/Pinpoint_APM)

**E-mail** : [roy.kim@navercorp.com](mailto:roy.kim@navercorp.com), [hyungil.jeong@navercorp.com](mailto:hyungil.jeong@navercorp.com)

