CommunityOverCode

THE ASF CONFERENCE

Web容器可观测最佳实践

陈承 高级开发工程师

THE APACHE SOFTWARE FOUNDATION

CONTENTS

1.传统的Web容器观测方案

2.基于可观测技术的Web容器观测方案

3.Web容器可观测未来发展方向

CommunityOverCode

THE ASF CONFERENCE

传统的Web容器可观测方案

THE APACHE SOFTWARE FOUNDATION

传统的Web容器监控方案

基于日志

tomcat manager

LambdaProbe(psi-probe)

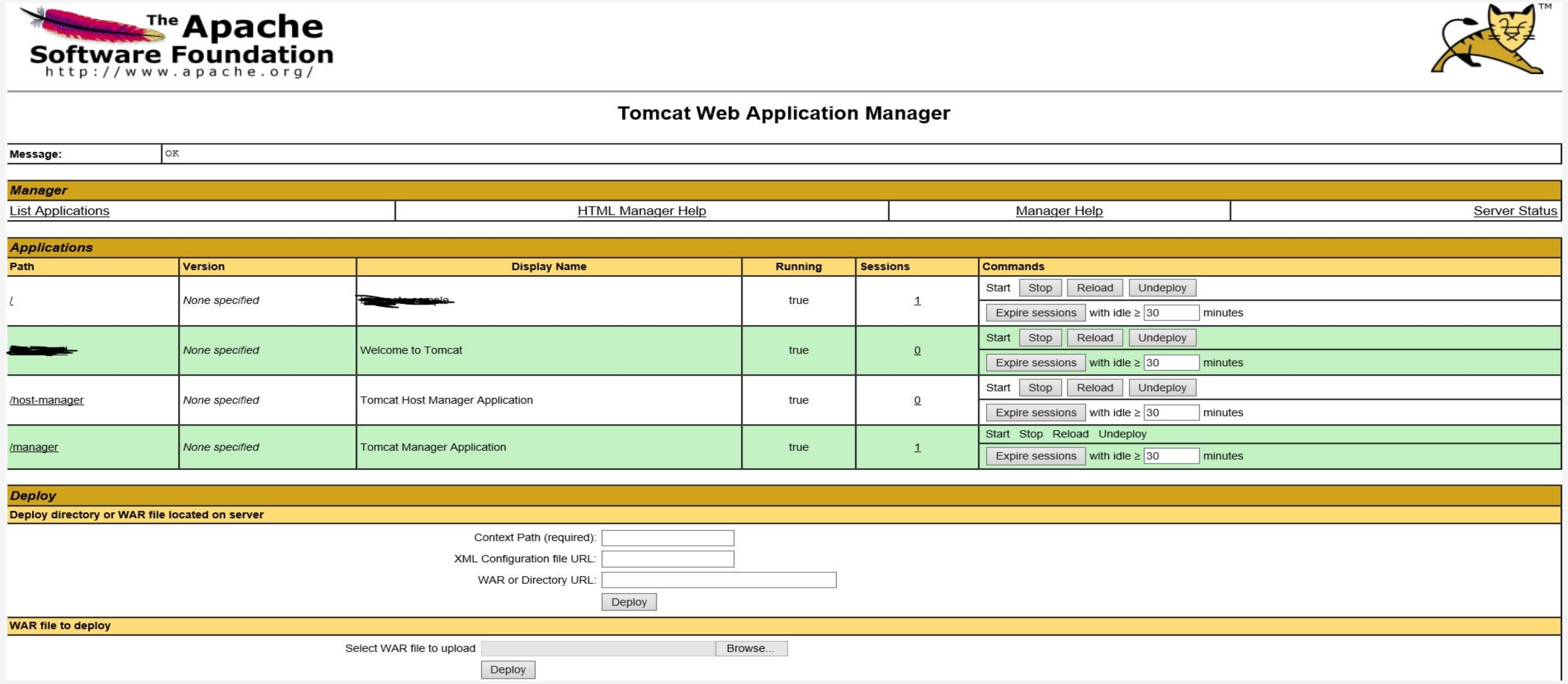
缺点

体系不完善

- 单点监控，无集群解决方案
- 没有历史数据

功能单一

- 无法多维分析
- 无法进行根因定位
- 无法创建报警


The Apache Software Foundation
http://www.apache.org/

Tomcat Web Application Manager

Message:	OK

Manager

List Applications	HTML Manager Help	Manager Help	Server Status

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	*None specified*	[redacted]	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
[redacted]	*None specified*	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	*None specified*	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	*None specified*	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

Deploy

Deploy directory or WAR file located on server

Context Path (required):

XML Configuration file URL:

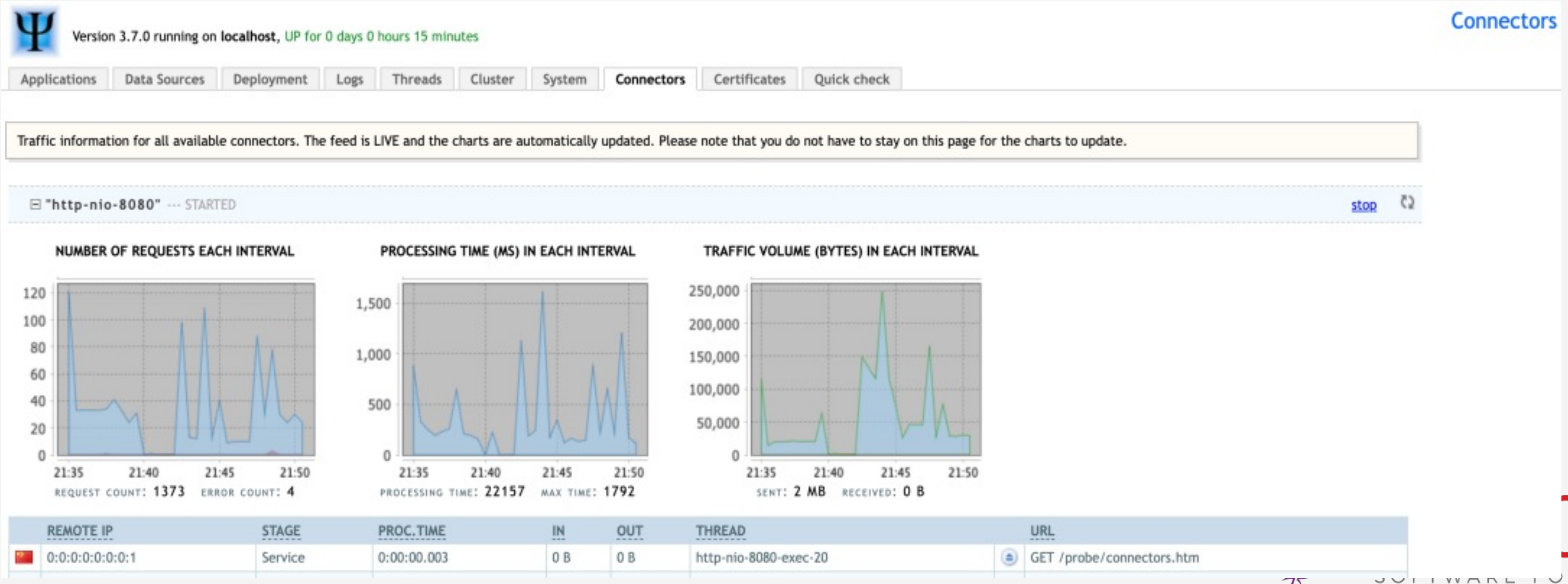
WAR or Directory URL:

Deploy

WAR file to deploy

Select WAR file to upload Browse...

Deploy


Applications
Data Sources
Deployment
Logs
Threads
Cluster
System
Connectors
Certificates
Quick check
stop
NUMBER OF REQUESTS EACH INTERVAL
120
100
80
60
40
20
0
21:35
21:40
21:45
21:50
PROCESSING TIME (MS) IN EACH INTERVAL
1,500
1,000
500
0
21:35
21:40
21:45
21:50
TRAFFIC VOLUME (BYTES) IN EACH INTERVAL
250,000
200,000
150,000
100,000
50,000
0
21:35
21:40
21:45
21:50

Version 3.7.0 running on **localhost**, UP for 0 days 0 hours 15 minutes

Connectors

Traffic information for all available connectors. The feed is LIVE and the charts are automatically updated. Please note that you do not have to stay on this page for the charts to update.

⊟ **"http-nio-8080"** --- STARTED

REQUEST COUNT: **1373** ERROR COUNT: **4**

PROCESSING TIME: **22157** MAX TIME: **1792**

SENT: **2 MB** RECEIVED: **0 B**

REMOTE IP	STAGE	PROC.TIME	IN	OUT	THREAD	URL
0:0:0:0:0:0:0:1	Service	0:00:00.003	0 B	0 B	http-nio-8080-exec-20	GET /probe/connectors.htm

CommunityOverCode

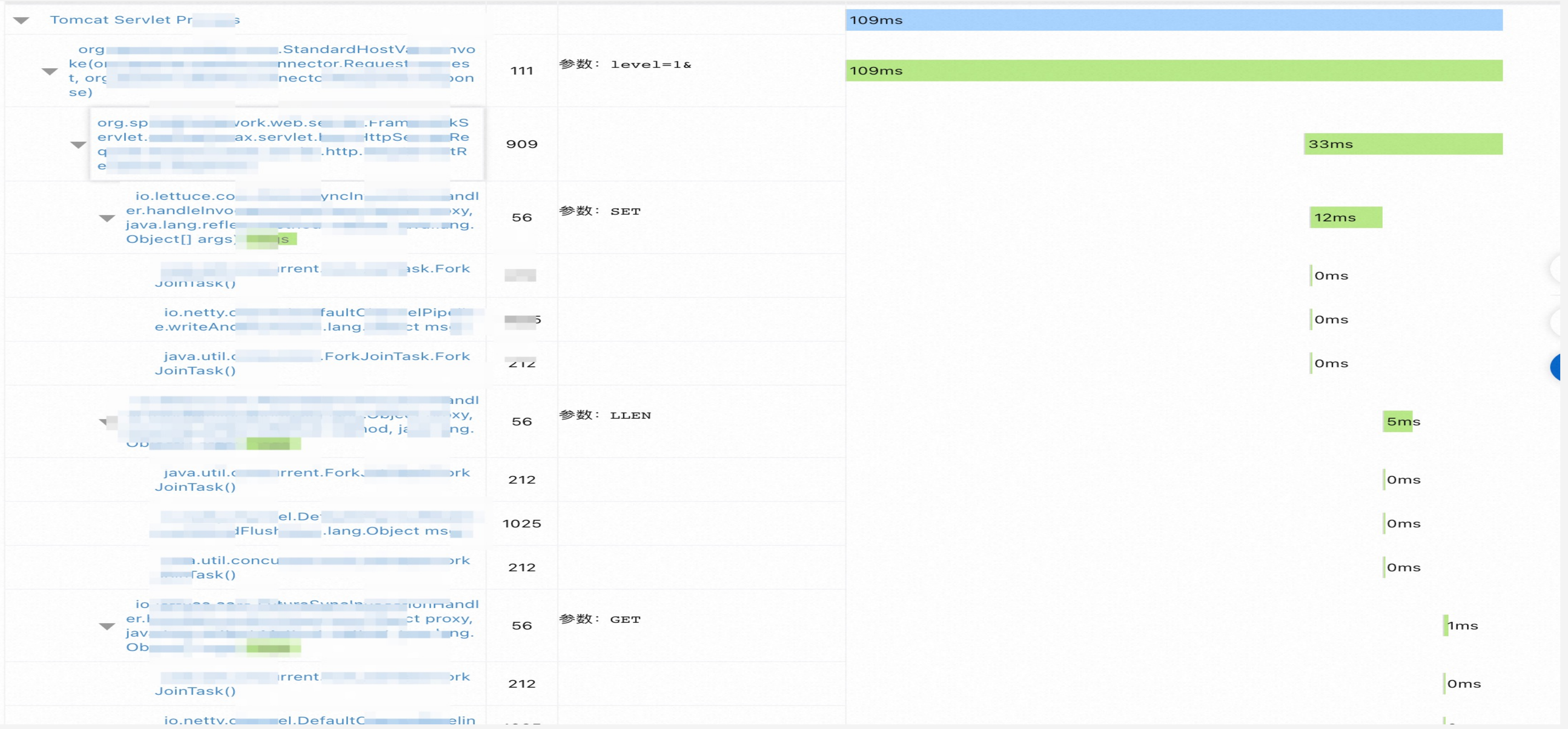
THE ASF CONFERENCE

基于可观测技术的Web容器监控方案

THE APACHE SOFTWARE FOUNDATION

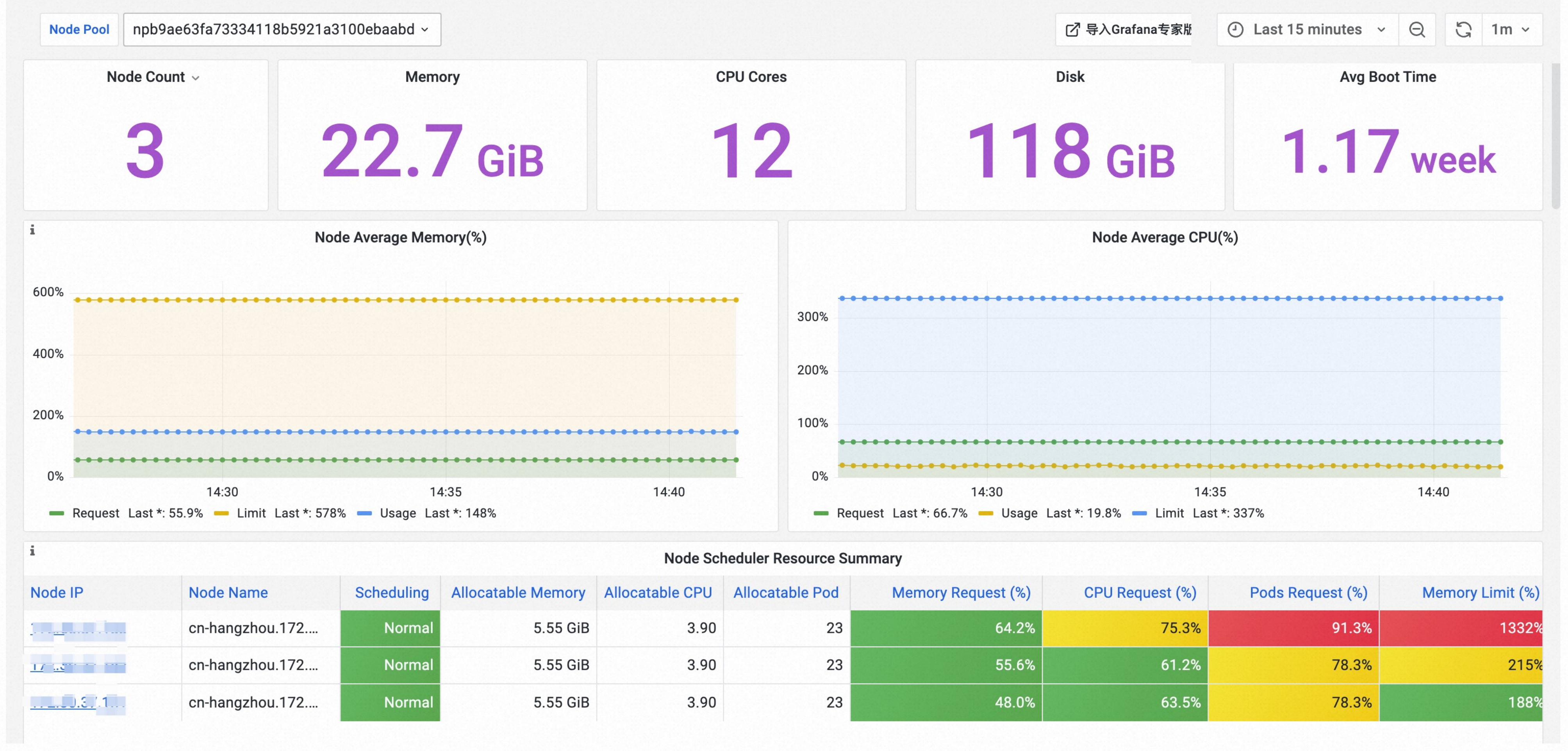
基本概念

trace



Tomcat Servlet Pr[redacted]s			109ms
org[redacted].StandardHostV[redacted]nvoke(or[redacted]nnector.Request[redacted]est, org[redacted]necto[redacted]onse)	111	参数：level=1&	109ms
org.sp[redacted]vork.web.se[redacted].Fram[redacted]kServlet.[redacted]ax.servlet.[redacted]ttpSe[redacted]Req[redacted].http.[redacted]tR e[redacted]	909		33ms
io.lettuce.co[redacted]yncIn[redacted]andler.handleInvo[redacted]xy, java.lang.refle[redacted]ng.Object[] args)[redacted]	56	参数：SET	12ms
[redacted]rrent.[redacted]ask.ForkJoinTask()	[redacted]		0ms
io.netty.c[redacted]faultC[redacted]elPipe[redacted]e.writeAnd[redacted].lang.[redacted]ct ms[redacted]	[redacted]		0ms
java.util.c[redacted].ForkJoinTask.ForkJoinTask()	212		0ms
[redacted]andl[redacted]xy, [redacted]hod, ja[redacted]ng.Ob[redacted]	56	参数：LLEN	5ms
java.util.c[redacted]rrent.Fork.[redacted]ork JoinTask()	212		0ms
[redacted]el.De[redacted]Flush[redacted].lang.Object ms[redacted]	1025		0ms
[redacted].util.concu[redacted]ork[redacted]Task()	212		0ms
io[redacted]onHandler.[redacted]ct proxy, jav[redacted]ng.Ob[redacted]	56	参数：GET	1ms
[redacted]rrent[redacted]ork JoinTask()	212		0ms
io.netty.c[redacted]el.DefaultC[redacted]elin			

metrics



Node Pool: npb9ae63fa73334118b5921a3100ebaabd

导入Grafana专家版
Last 15 minutes
1m

Node Count: 3

Memory: 22.7 GiB

CPU Cores: 12

Disk: 118 GiB

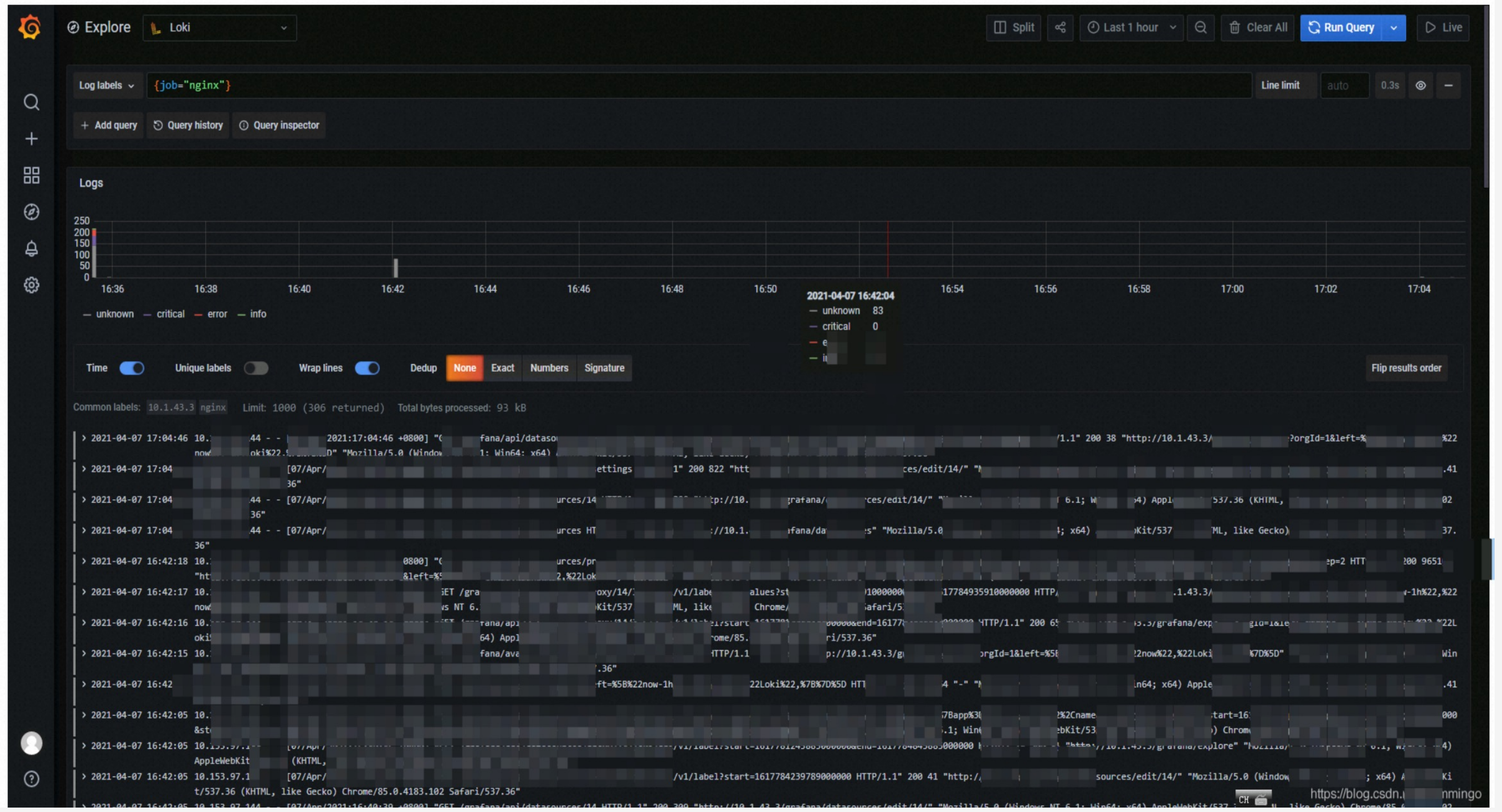
Avg Boot Time: 1.17 week

Node Average Memory(%)
600%
400%
200%
0%
14:30
14:35
14:40
Request Last *: 55.9%
Limit Last *: 578%
Usage Last *: 148%
Node Average CPU(%)
300%
200%
100%
0%
14:30
14:35
14:40
Request Last *: 66.7%
Usage Last *: 19.8%
Limit Last *: 337%

Node Scheduler Resource Summary

Node IP	Node Name	Scheduling	Allocatable Memory	Allocatable CPU	Allocatable Pod	Memory Request (%)	CPU Request (%)	Pods Request (%)	Memory Limit (%)
[redacted]	cn-hangzhou.172....	Normal	5.55 GiB	3.90	23	64.2%	75.3%	91.3%	1332%
[redacted]	cn-hangzhou.172....	Normal	5.55 GiB	3.90	23	55.6%	61.2%	78.3%	215%
[redacted]	cn-hangzhou.172....	Normal	5.55 GiB	3.90	23	48.0%	63.5%	78.3%	188%

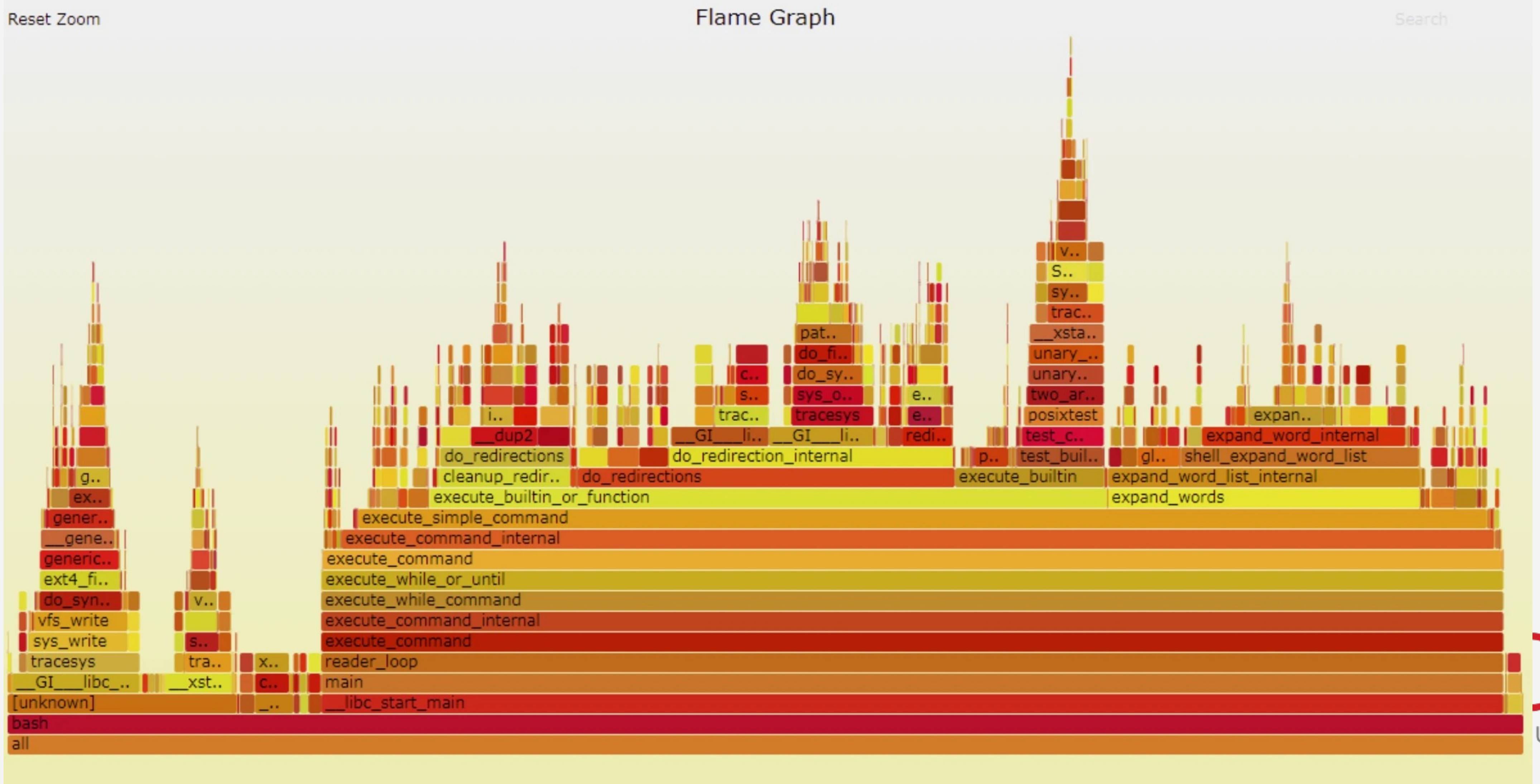
log



Explore
Loki
Split
Last 1 hour
Clear All
Run Query
Live
Log labels
{job="nginx"}
Line limit
auto
+ Add query
Query history
Query inspector
Logs
250
200
150
100
50
0
16:36
16:38
16:40
16:42
16:44
16:46
16:48
16:50
16:54
16:56
16:58
17:00
17:02
17:04
unknown
critical
error
info
2021-04-07 16:42:04
unknown 83
critical 0
Time
Unique labels
Wrap lines
Dedup
None
Exact
Numbers
Signature
Flip results order

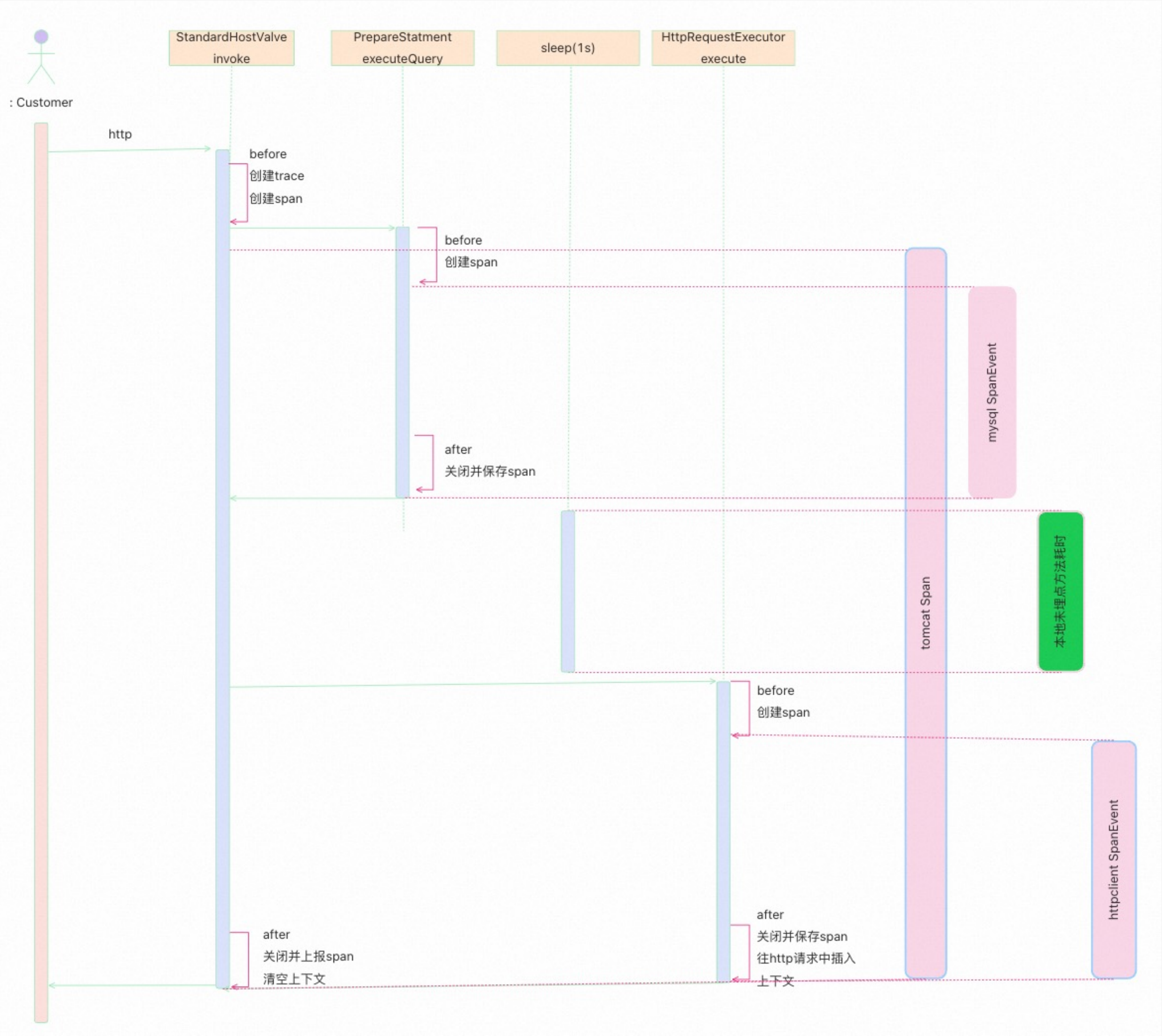
Common labels: 10.1.43.3 nginx Limit: 1000 (306 returned) Total bytes processed: 93 kB

profiling



基本原理

Trace 原理


: Customer
StandardHostValve invoke
PrepareStatment executeQuery
sleep(1s)
HttpRequestExecutor execute
http
before
创建trace
创建span
before
创建span
after
关闭并保存span
mysql SpanEvent
本地未埋点方法耗时
tomcat Span
before
创建span
httpclient SpanEvent
after
关闭并保存span
往http请求中插入
上下文
after
关闭并上报span
清空上下文

Metrics 原理

业界的标准：Prometheus

指标模型：单指标，多维度

指标类型：
- Counter
- Gauge
- Histogram
- Summary

关键概念：
- 时间点
- 时间线
- promQL

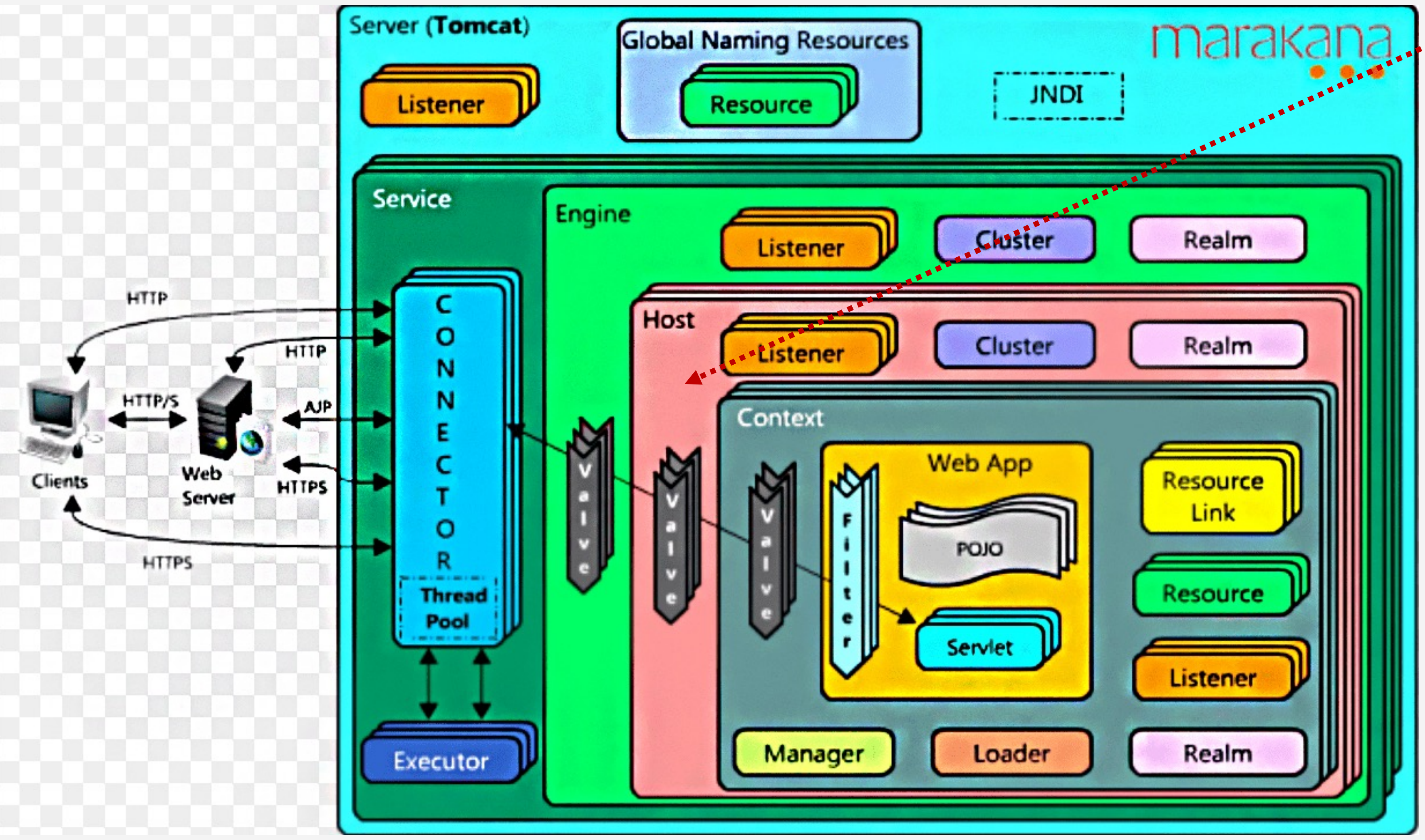
Trace最佳实践

传统WEB服务器埋点关键点

- 保证用户的业务代码都能获取上下文

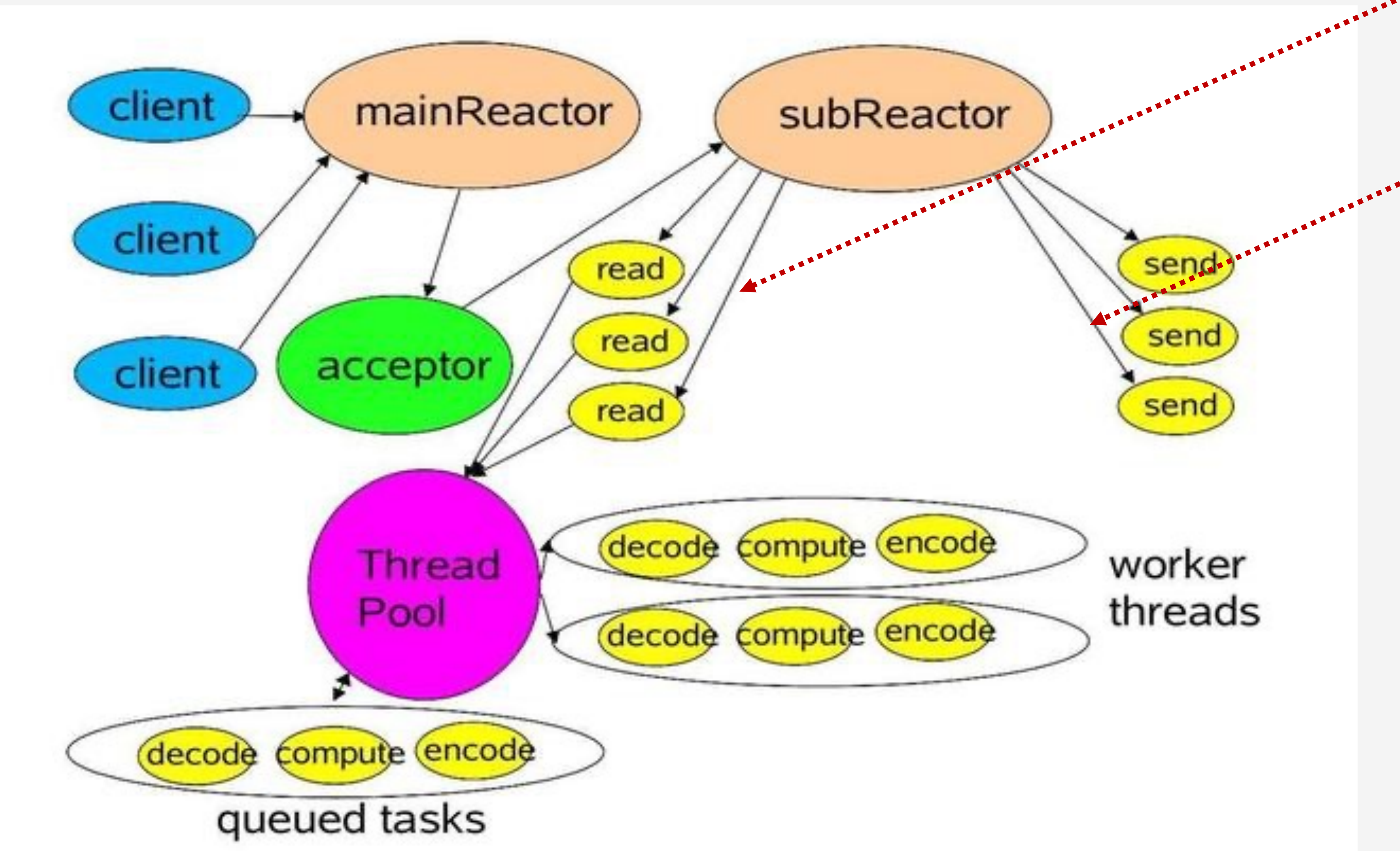
新型Web服务端技术埋点关键点

- 保证耗时统计准确



Server (Tomcat)
Listener
Global Naming Resources
Resource
JNDI
marakana
Service
Engine
Listener
Cluster
Realm
Host
Listener
Cluster
Realm
Context
Web App
POJO
Servlet
Filter
Valve
Valve
Valve
Resource Link
Resource
Listener
Manager
Loader
Realm
CONNECTOR
Thread Pool
Executor
HTTP
HTTP
HTTP/S
AJP
HTTPS
HTTPS
Clients
Web Server

创建、关闭span



client
client
client
mainReactor
subReactor
acceptor
read
read
read
send
send
send
Thread Pool
decode
compute
encode
decode
compute
encode
worker threads
decode
compute
encode
queued tasks

创建span

关闭span

Metrics最佳实践

端侧预聚合指标

- 指标精确不受采样影响
- 上报数据量较少

date	spanName	duration	select spanName, count(1) as count, sum(duration) as totalDuration group by spanName		
12:00:01	/helloword/1	32	count	totalDuration	spanName
12:00:08	/helloword/1	57	5	168	/helloword/1
12:00:08	/helloword/1	27			
12:00:10	/helloword/1	31			
12:00:14	/helloword/1	22			
12:00:15	/helloword/1	32	3	111	/helloword/1
12:00:17	/helloword/2	57			
12:00:19	/helloword/4	27	2	88	/helloword/2
12:00:21	/helloword/2	31			
12:00:23	/helloword/1	22	1	27	/helloword/4
12:00:26	/helloword/5	32			
12:00:28	/helloword/1	57	1	32	/helloword/5

关键指标

- 请求数
- 请求大小（分位数）
- 响应大小（分位数）
- 耗时（分位数）
- 活跃请求数
- 线程池指标

关键维度

- 接口名
- 上游接口名
- 状态码

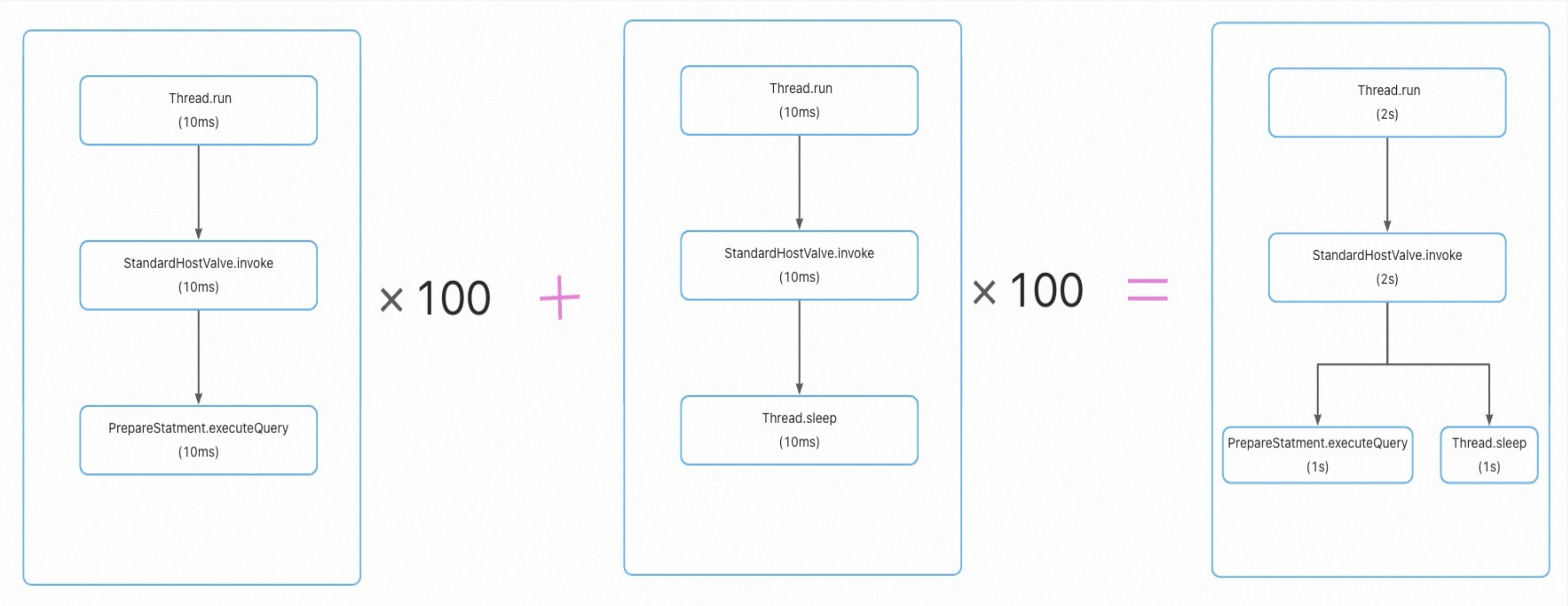
难点

- 维度收敛
  - 根据org.springframework.web.servlet.HandlerMapping #BEST_MATCHING_PATTERN_ATTRIBUTE收敛
  - SCG 后收敛
- 精准指标
  - 按照时间分桶
- 准确的分位数
  - DDSketch 算法（算法介绍）

Profiling 基本原理

Profiling常见技术：perf、 Async-profiler、 JFR

CPU火焰图：



Thread.run (10ms)
StandardHostValve.invoke (10ms)
PrepareStatment.executeQuery (10ms)

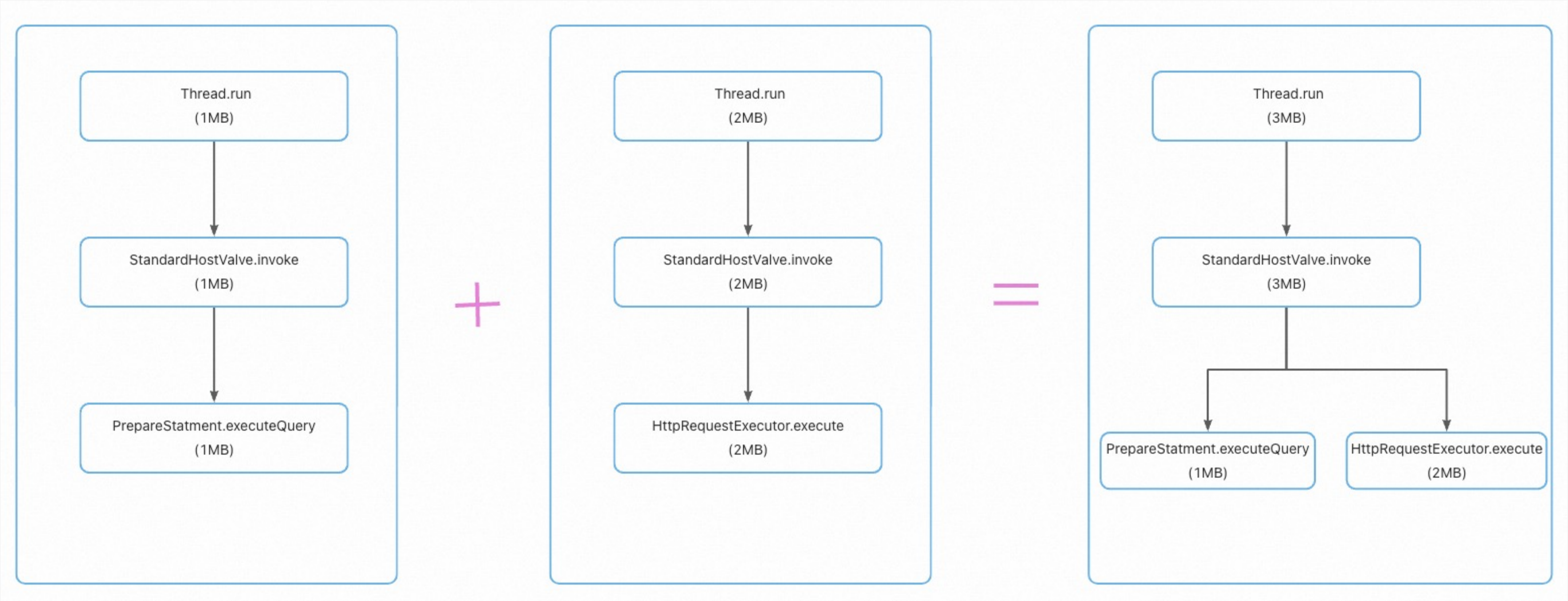
× 100 +

Thread.run (10ms)
StandardHostValve.invoke (10ms)
Thread.sleep (10ms)

× 100 =

Thread.run (2s)
StandardHostValve.invoke (2s)
PrepareStatment.executeQuery (1s)
Thread.sleep (1s)

内存火焰图：



Thread.run (1MB)
StandardHostValve.invoke (1MB)
PrepareStatment.executeQuery (1MB)

+

Thread.run (2MB)
StandardHostValve.invoke (2MB)
HttpRequestExecutor.execute (2MB)

=

Thread.run (3MB)
StandardHostValve.invoke (3MB)
PrepareStatment.executeQuery (1MB)
HttpRequestExecutor.execute (2MB)

THE APACHE® SOFTWARE FOUNDATION

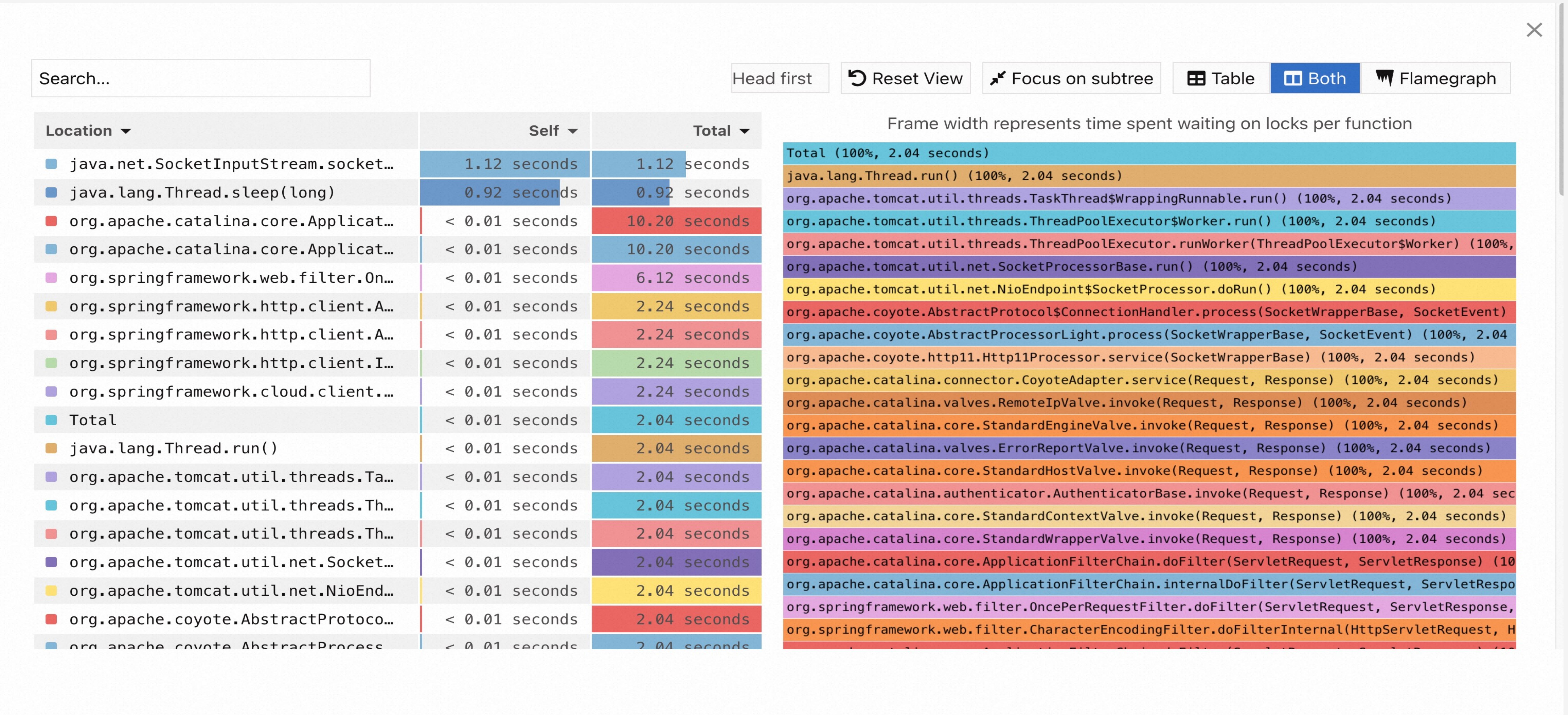
Profiling最佳实践

Stack关联Trace上下文、Metrics维度、业务上下文

- traceId
- spanId
- 接口名
- userId

效果

- 可以看到不同接口的资源占用、驱动应用调优
- 可以看到某一次调用的资源占用，驱动异常调用识别
- 可以看到不同用户对我们系统的资源占用，驱动异常用户识别



Location	Self	Total
java.net.SocketInputStream.socket…	1.12 seconds	1.12 seconds
java.lang.Thread.sleep(long)	0.92 seconds	0.92 seconds
org.apache.catalina.core.Applicat…	< 0.01 seconds	10.20 seconds
org.apache.catalina.core.Applicat…	< 0.01 seconds	10.20 seconds
org.springframework.web.filter.On…	< 0.01 seconds	6.12 seconds
org.springframework.http.client.A…	< 0.01 seconds	2.24 seconds
org.springframework.http.client.A…	< 0.01 seconds	2.24 seconds
org.springframework.http.client.I…	< 0.01 seconds	2.24 seconds
org.springframework.cloud.client.…	< 0.01 seconds	2.24 seconds
Total	< 0.01 seconds	2.04 seconds
java.lang.Thread.run()	< 0.01 seconds	2.04 seconds
org.apache.tomcat.util.threads.Ta…	< 0.01 seconds	2.04 seconds
org.apache.tomcat.util.threads.Th…	< 0.01 seconds	2.04 seconds
org.apache.tomcat.util.threads.Th…	< 0.01 seconds	2.04 seconds
org.apache.tomcat.util.net.Socket…	< 0.01 seconds	2.04 seconds
org.apache.tomcat.util.net.NioEnd…	< 0.01 seconds	2.04 seconds
org.apache.coyote.AbstractProtoco…	< 0.01 seconds	2.04 seconds
org.apache.coyote.AbstractProcess	< 0.01 seconds	2.04 seconds

Frame width represents time spent waiting on locks per function

```
Total (100%, 2.04 seconds)
java.lang.Thread.run() (100%, 2.04 seconds)
org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run() (100%, 2.04 seconds)
org.apache.tomcat.util.threads.ThreadPoolExecutor$Worker.run() (100%, 2.04 seconds)
org.apache.tomcat.util.threads.ThreadPoolExecutor.runWorker(ThreadPoolExecutor$Worker) (100%,
org.apache.tomcat.util.net.SocketProcessorBase.run() (100%, 2.04 seconds)
org.apache.tomcat.util.net.NioEndpoint$SocketProcessor.doRun() (100%, 2.04 seconds)
org.apache.coyote.AbstractProtocol$ConnectionHandler.process(SocketWrapperBase, SocketEvent)
org.apache.coyote.AbstractProcessorLight.process(SocketWrapperBase, SocketEvent) (100%, 2.04
org.apache.coyote.http11.Http11Processor.service(SocketWrapperBase) (100%, 2.04 seconds)
org.apache.catalina.connector.CoyoteAdapter.service(Request, Response) (100%, 2.04 seconds)
org.apache.catalina.valves.RemoteIpValve.invoke(Request, Response) (100%, 2.04 seconds)
org.apache.catalina.core.StandardEngineValve.invoke(Request, Response) (100%, 2.04 seconds)
org.apache.catalina.valves.ErrorReportValve.invoke(Request, Response) (100%, 2.04 seconds)
org.apache.catalina.core.StandardHostValve.invoke(Request, Response) (100%, 2.04 seconds)
org.apache.catalina.authenticator.AuthenticatorBase.invoke(Request, Response) (100%, 2.04 sec
org.apache.catalina.core.StandardContextValve.invoke(Request, Response) (100%, 2.04 seconds)
org.apache.catalina.core.StandardWrapperValve.invoke(Request, Response) (100%, 2.04 seconds)
org.apache.catalina.core.ApplicationFilterChain.doFilter(ServletRequest, ServletResponse) (10
org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ServletRequest, ServletRespo
org.springframework.web.filter.OncePerRequestFilter.doFilter(ServletRequest, ServletResponse,
org.springframework.web.filter.CharacterEncodingFilter.doFilterInternal(HttpServletRequest, H
```

Web容器可观测未来发展方向

Profiling最佳实践

性能：

- 可预期可管理的性能开销

功能

- 提供更符合用户预期的采样行为
- tomcat 10的支持
- GraalVM场景下Web服务器的埋点

智能

- 智能计算逻辑下沉

CommunityOverCode

THE ASF CONFERENCE

Thanks

陈承 高级开发工程师

wu.cc@alibaba-inc.com

WWW.COMMUNITYOVERCODE.ORG

THE APACHE SOFTWARE FOUNDATION