



# 1、从零搭建SkyWalking（9.4.0版本），来监控收集微服务调用链路数据

## （0）开发手册

- <https://skywalking.apache.org/docs/main/v9.4.0/readme/>

## （1）使用SkyWalking的原因

- 项目采用微服务的架构，服务很多，人工监控是不可能的，项目的访问量很大，想通过日志查找某个方法中性能的问题也是非常困难的，所以skywalking应运而生。

## （2）SkyWalking是什么

- + ::
- <https://skywalking.apache.org/>
  - 2015年由个人吴晟（华为开发者）开源的一套专门为分布式微服务架构、云原生架构、基于容器（Docker、K8S、Mesos）的微服务架构而设计的性能监视检测工具（接口链路数据追踪，分析，告警），2017年此项目加入了Apache基金会（ASF），被ASF的孵化器进行孵化，实力可见一斑。也称为APM（Application Performance Management）。

Apache孵化器有一整套针对开源软件的治理机制，构建一个公开透明，健康发展的开源社区，在质量,法律方面有保障，所以可以获得更多的关注，也可以吸引更多的用户以及开发者参与，持续更新，软件的价值就会上升，把项目捐赠给 ASF 还是一个很好的选择。

- 一句话：是一款分布式系统的性能监视工具，提供了链路追踪（Tracing），服务网格遥测分析（Analysis），度量指标聚合（Metric）、可视化一体化的解决方案。

## （3）SkyWalking的架构图

- 整个架构分成4部分

上部分：java agent探针

（1）目前java agent探针支持收集skywalking，zikpin，jaeger等提供的Tracing数据。

（2）skywalking agent（java agent for skywalking，支持收集skywalking Tracing数据的java agent）负责从boot应用中探测收集接口链路信息（skywalking Tracing数据），发送给skywalking oap服务器。



中部分：skywalking oap服务

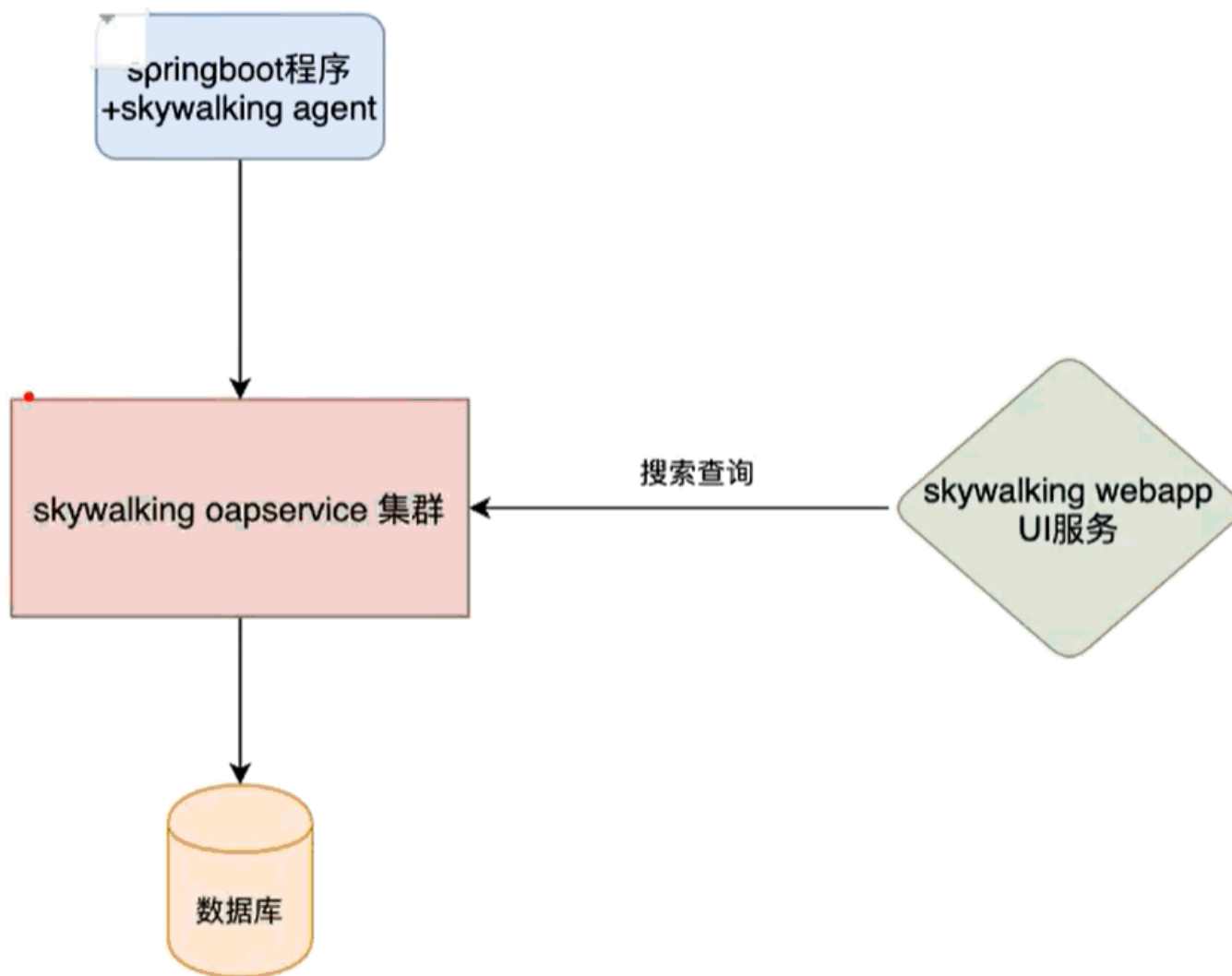
（1）负责接收skywalking agent发送过来的skywalking Tracing数据，然后进行分析（Analysis），存储到外部存储器（Storage etcd），也就是存储到下图中的数据库，最终提供查询数据库的功能API。

下部分：etcd存储（数据库）

（1）负责存储skywalking oap服务发送过来的skywalking Tracing 数据，目前支持ES，MySQL，ShardingSphere，TiDB，H2等多种存储器，SkyWalking 开发团队自己的生产环境采用 ES 为主，所以最好使用ES做存储。（案例演示就使用H2内存型数据库，无需任何配置）

右部分：SkyWalking webapp UI服务

（1）用来调用skywalking oap提供的查询API，来展示链路追踪，日志，拓扑图，告警等监控信息。



#### (4) 下载

##### (4-1) 官网下载地址

- <https://skywalking.apache.org/downloads/>

## Foundations



### SkyWalking APM

SkyWalking is an Observability Analysis Platform and Application Performance Management system.

Source

源码

Distribution

安装包

包括oap和webapp ui



### Rocketbot UI

SkyWalking's primary UI. All source codes have been included in the main repo release.

Included in the main repo release



### SkyWalking Website

All source codes of <https://skywalking.apache.org>

Deployed

## Agents

提供不同语言的agent，  
这里使用 java版本的agent探针技术来收集追踪信息



### Java Agent

The Java Agent for Apache SkyWalking, which provides the native tracing/metrics/logging/event/profiling abilities for Java projects.

Source

Distribution



### Python Agent

The Python Agent for Apache SkyWalking, which provides the native tracing/metrics/logging/profiling abilities for Python projects.

Source

Distribution



### Go Agent

The Go Agent for Apache SkyWalking provides the native tracing/metric abilities for Golang projects.

Source

Di

### ( 4-2 ) 下载SkyWalking APM

- 下载完解压即可

## Foundations



### SkyWalking APM

SkyWalking is an Observability Analysis Platform and Application Performance Management system.

Source ▾

Distribution ▾

Include ▾

v9.4.0 | Mar. 12th, 2023

[\[tar\]](#) [\[asc\]](#) [\[sha512\]](#)



### Rocket

SkyWalk  
have be



## COMMUNITY-LED DEVELOPMENT "THE APACHE WAY"

Projects ▾

People ▾

Community ▾

License ▾

We suggest the following site for your download:

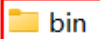
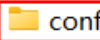


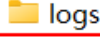

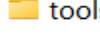
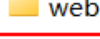
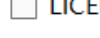
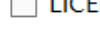
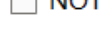
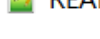
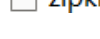
<https://dlcdn.apache.org/skywalking/9.4.0/apache-skywalking-apm-9.4.0.tar.gz>

Alternate download locations are suggested below.

It is essential that you [verify the integrity](#) of the downloaded file using the PGP signature ([.asc](#) file) or a hash ([.md5](#) or [.sha\\*](#) file).

- 解压目录介绍
  - 整体介绍

> 此电脑 > Data (D:) > software > apache-skywalking-apm-9.4.0 > apache-skywalking-apm-bin

	名称	修改日期
	 bin 启动oap 服务和webapp ui服务	2023/6/5 15:27
★	 config 配置	2023/6/5 15:27
★	 config-examples	2023/6/5 15:27
★	 licenses	2023/6/5 15:27
★	 logs 启动日志	2023/6/7 9:39
★	 oap-libs oap的依赖jar	2023/6/6 15:31
★	 tools	2023/6/5 15:27
★	 webapp web app ui的配置和依赖jar	2023/6/5 15:27
★	 LICENSE	2022/12/12 3:18
★	 LICENSE.tpl	2022/12/12 3:18
★	 NOTICE	2022/12/12 3:18
	 README.txt	2022/12/12 3:18
	 zipkin-LICENSE	2022/12/12 3:18

- bin文件夹：是oap、webapp ui的启动命令

Data (D:) > software > apache-skywalking-apm-9.4.0 > apache-skywalking-apm-bin > bin					在 bin 中搜索
名称	修改日期	类型	大小		
oapService.bat	2022/12/12 3:18	Windows 批处理文件	2 KB		
oapService.sh	2022/12/12 3:18	Shell Script	2 KB		
oapServiceInit.bat	2022/12/12 3:18	Windows 批处理文件	2 KB		
oapServiceInit.sh	2022/12/12 3:18	Shell Script	2 KB		
oapServiceNoInit.bat	2022/12/12 3:18	Windows 批处理文件	2 KB		
oapServiceNoInit.sh	2022/12/12 3:18	Shell Script	2 KB		
startup.bat	2022/12/12 3:18	Windows 批处理文件	1 KB		
startup.sh	2022/12/12 3:18	Shell Script	1 KB		
webappService.bat	2022/12/12 3:18	Windows 批处理文件	2 KB		
webappService.sh	2022/12/12 3:18	Shell Script	2 KB		

点击启动后会启动两个服务，一个是skywalking-oap-server服务，一个是skywalking-web-ui服务  
(1) skywalking-oap-server服务：

暴露11800和12800端口，11800端口用来接收agent收集的监控数据的端口，12800端口用来提供webapp ui请求数据的端口。

(2) skywalking-web-ui服务：

暴露8080端口，用来提供请求监控页面的端口。

#### ○ webapp文件夹

- application.yml中配置了请求web ui页面的8080端口和配置了webapp ui界面请求oap服务查询数据展示的12800端口。端口可自定义修改，请求webapp ui页面的端口我改成了8088。

› Data (D:) › software › apache-skywalking-apm-9.4.0 › apache-skywalking-apm-bin › webapp

名称	修改日期	类型
 application.yml	2023/6/5 23:01	YML 文件
 log4j2.xml	2022/12/12 3:18	XML 文件
 skywalking-webapp.jar	2022/12/12 3:18	Executable

```
serverPort: ${SW_SERVER_PORT:-8080}  
# Comma seperated list of OAP addresses.  
oapServices: ${SW_OAP_ADDRESS:-http://localhost:12800}  
zipkinServices: ${SW_ZIPKIN_ADDRESS:-http://localhost:9412}
```

如果oap是集群，可数组，逗号分割



- config文件夹：告警规则配置，数据库配置



Data (D:) > software > apache-skywalking-apm-9.4.0 > apache-skywalking-apm-bin > config >				▼	↺
	名称	修改日期	类型		
	envoy-metrics-rules	2023/6/5 15:27	文件夹		
★	lal	2023/6/5 15:27	文件夹		
★	log-mal-rules	2023/6/5 15:27	文件夹		
★	meter-analyzer-confi	2023/6/5 15:27	文件夹		
★	oal	2023/6/5 15:27	文件夹	配置默认或自定义度量指标（包括服务，实例，端点的指标）	
★	openapi-definitions	2023/6/5 15:27	文件夹		
★	otel-rules	2023/6/5 15:27	文件夹		
★	telegraf-rules	2023/6/5 15:27	文件夹		
★	ui-initialized-templates	2023/6/5 15:27	文件夹		
★	zabbix-rules	2023/6/5 15:27	文件夹	配置告警规则：内置的告警规则中引用度量指标，度量指标达到阈值后，进行告警	
★	alarm-settings.yml	2023/6/7 9:54	YML 文件		
	application.yml	2023/6/7 0:47	YML 文件	配置使用哪种数据库	
	component-libraries.yml	2022/12/12 3:18	YML 文件		

- logs文件夹：oap和webapp ui服务启动时，会生成对应服务的启动日志文件

› Data (D:) › software › apache-skywalking-apm-9.4.0 › apache-skywalking-apm-bin › logs

名称	修改日期
 skywalking-oap-server.log	2023/6/7 10:45
 skywalking-webapp.log	2023/6/7 9:40

#### ( 4-3 ) 下载Java Agent

- 老版本的Agent是集成在APM安装包里面，都在一块，而这里新版本的Agent要单独下载

## Agents



### Java Agent

The Java Agent for Apache SkyWalking, which provides the native tracing/metrics/logging/event/profiling abilities for Java projects.

Source ▾



### Python

The Pyth which pr tracing/r Python p

Source ▾

Distribution ▾

v8.16.0 | Jun. 2nd, 2023

[\[tar\]](#) [\[asc\]](#) [\[sha512\]](#)

- boot应用中接口链路数据要想被追踪，需要引入Java Agent for skywalking（skywalking-java-agent）

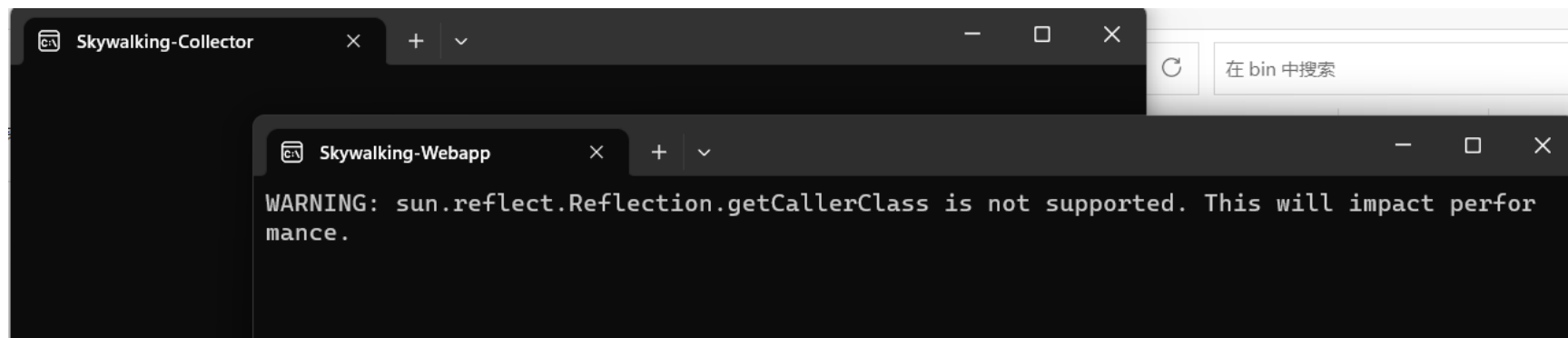
此电脑 > Data (D:) > software > apache-skywalking-java-agent-8.16.0 > skywalking-agent >

	名称	修改日期
	activations	2023/6/5 14:26
✦	bootstrap-plugins	2023/6/5 14:26
✦	config	2023/6/5 14:26
✦	licenses	2023/6/5 14:26
✦	logs	2023/6/5 16:03
✦	optional-plugins	2023/6/5 14:26
✦	optional-reporter-plugins	2023/6/5 14:26
✦	plugins	2023/6/5 14:26
✦	LICENSE	2023/5/31 23:37
✦	NOTICE	2023/5/31 23:37
✦	skywalking-agent.jar	2023/5/31 23:37

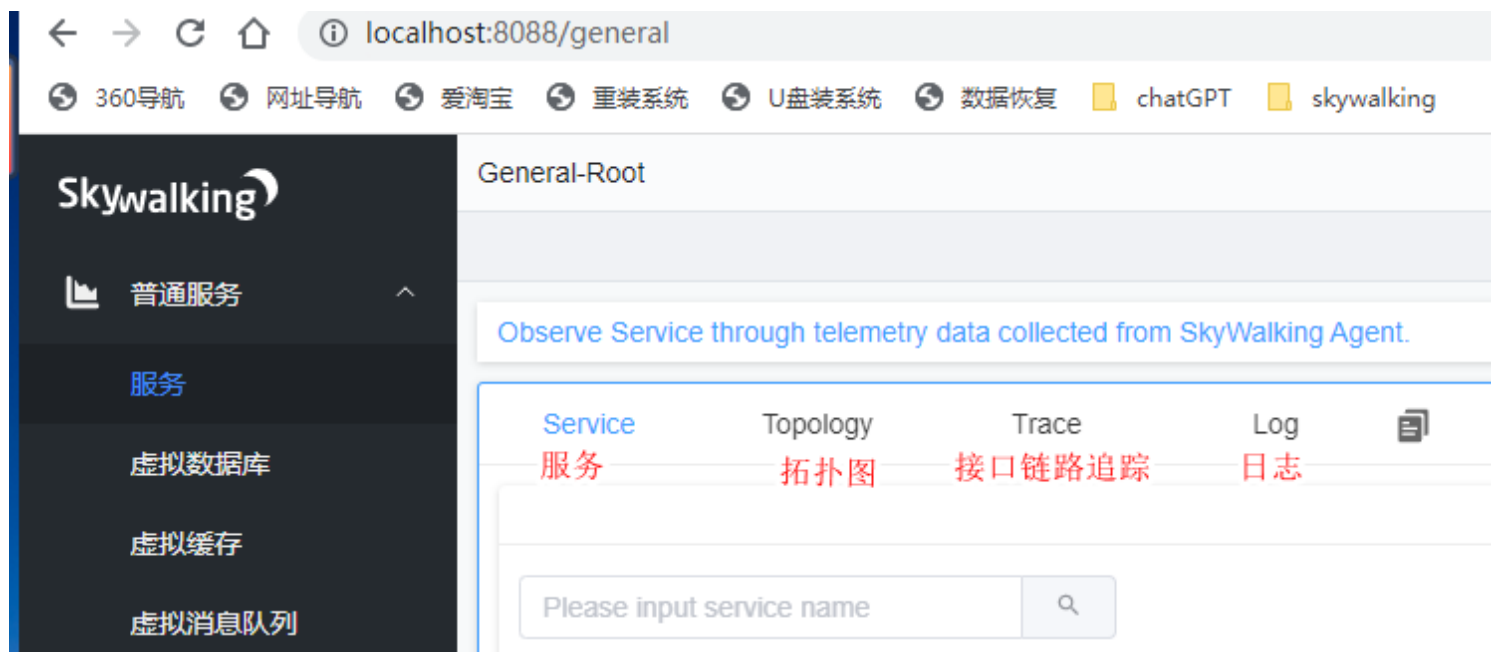
## （5）启动服务

- 去bin文件夹中双击startup.bat（window版本）启动oap、webapp ui服务，然后查看logs文件夹中启动日志是否正常





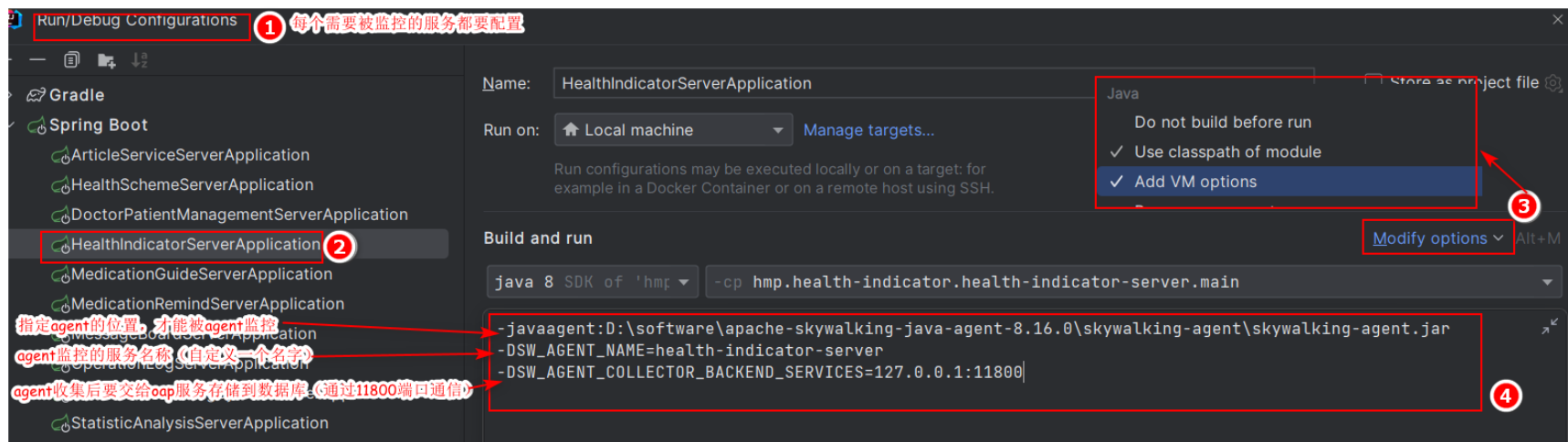
- 访问http://localhost:8088打开webapp ui界面



(6) boot应用配置agent, 进行监控

(6-1) 接口调用链路的监控

- 需要在IDEA的boot应用的Edit Configurations的VM Options选项中指定skywalking-agent.jar的路径，boot应用调用链路才能被agent监控收集（agent在boot项目编译期间侵入到源码中进行监控，源码级别无侵入）。

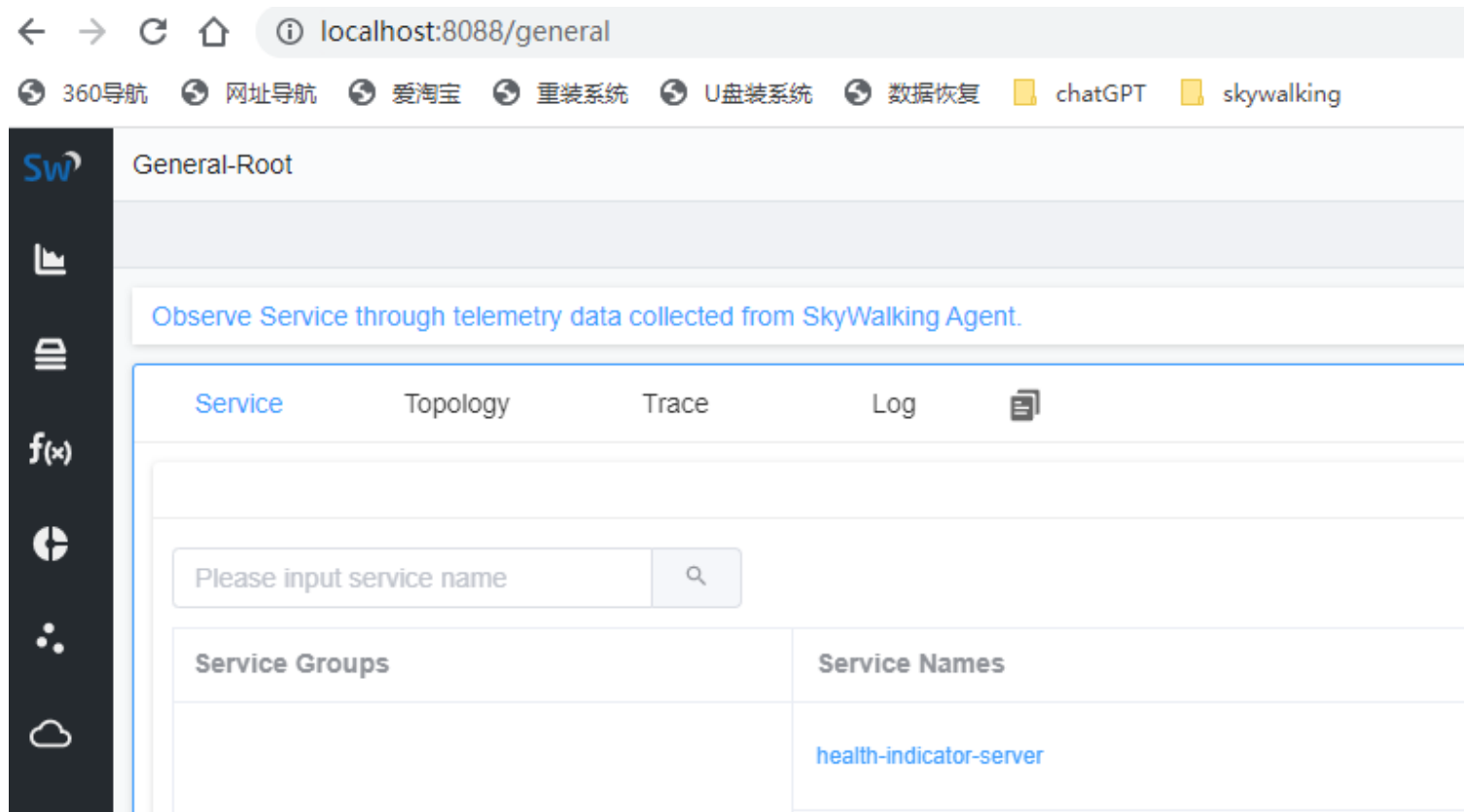


```
1 //java agent的jar包的路径
2 -javaagent:D:\software\apache-skywalking-java-agent-8.16.0\skywalking-agent\skywalking-agent.jar
3 //指定agent监控收集的服务名称，可自定义名称
4 -DSW_AGENT_NAME=health-indicator-server
5 //指定agent监控收集后，交给oap服务（11800端口通信），存储到数据库
6 -DSW_AGENT_COLLECTOR_BACKEND_SERVICES=127.0.0.1:11800
```

复制代码

java >

- 启动boot服务后，再次重新访问http://localhost:8088打开skywalking webapp ui界面，可以看到刚刚agent的配置生效了。
  - agent收集追踪数据交给oap服务（11800端口通信），oap服务存储到数据库。
  - 然后直接访问skywalking webapp ui页面（8088端口通信），页面上的数据是请求oap服务（12800端口通信）到数据库查到的。



## ( 6-2 ) 程序中输出日志的监控

- 因为boot应用默认使用的是logback框架来记录程序运行中的日志，打印到控制台或输出到文件中，所以可以将logback框架和skywalking整合，通过grpc远程调用的方式，将日志输出到skywalking 服务的数据库中。
  - 引入skywalking和logback的整合jar包

```
1 //https://mvnrepository.com/artifact/org.apache.skywalking/apm-toolkit-logback-1.x
2 implementation("org.apache.skywalking:apm-toolkit-logback-1.x")
```

复制代码

java >

- resources资源目录下创建logback.yml，整合skywalking
  - 具体可参考官网地址：<https://skywalking.apache.org/docs/main/v9.4.0/en/setup/backend/log-analyzer/>

← → ↻ 🏠 [skywalking.apache.org/docs/main/v9.4.0/en/setup/backend/log-analyzer/](https://skywalking.apache.org/docs/main/v9.4.0/en/setup/backend/log-analyzer/)

🌐 360导航 🌐 网址导航 🌐 爱淘宝 🌐 重装系统 🌐 U盘装系统 🌐 数据恢复 🟡 chatGPT 🟡 skywalking



Projects an

telemetry data report and query.

Version: v9.4.0 ▼

Welcome

Concepts and Designs >

Setup ▼

Quick Start >

Configuration Vocabulary

Advanced Setup >

Tracing >

Metrics >

Logging ▼

Log Collecting And Analysis

On Demand Pod Logs

Take the following fluent-bit config files as an example to set up Flue

- [fluent-bit.conf](#)

## OpenTelemetry

You can use OpenTelemetry Collector to transport the logs to SkyWal

## Java agent's toolkits

Java agent provides toolkits for [log4j](#), [log4j2](#), and [logback](#) to report l

[SkyWalking Satellite sidecar](#) is a recommended proxy/side that forwa

Java agent provides toolkits for [log4j](#), [log4j2](#), and [logback](#) to report l

Log framework config examples:

- [log4j1.x fileAppender](#)
- [log4j2.x fileAppender](#)
- [logback fileAppender](#)



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <configuration scan="true" scanPeriod="15 seconds">
3     <include resource="org/springframework/boot/logging/logback/defaults.xml"/>
4
5     <appender name="grpc-log"
6         class="org.apache.skywalking.apm.toolkit.log.logback.v1.x.log.GRPCLogClientAppender">
7         <encoder class="ch.qos.logback.core.encoder.LayoutWrappingEncoder">
8             <layout
9                 class="org.apache.skywalking.apm.toolkit.log.logback.v1.x.mdc.TraceIdMDCPatternLogbackLayout"
10             >
11                 <Pattern>%d{yyyy-MM-dd'T'HH:mm:ss.SSSXXX} [%X{tid}] [%thread] %-5level
12 %logger{36} -%msg%n</Pattern>
13             </layout>
14         </encoder>
15     </appender>
16
17     <appender name="CONSOLE_SKY" class="ch.qos.logback.core.ConsoleAppender">
18         <encoder class="ch.qos.logback.core.encoder.LayoutWrappingEncoder">
19             <charset>${CONSOLE_LOG_CHARSET}</charset>
20             <layout
21                 class="org.apache.skywalking.apm.toolkit.log.logback.v1.x.mdc.TraceIdMDCPatternLogbackLayout"
22             >
23                 <pattern>[%X{tid}] ${CONSOLE_LOG_PATTERN}</pattern>
24             </layout>
25         </encoder>
26     </appender>
27
28     <appender name="ASYNC" class="ch.qos.logback.classic.AsyncAppender">
29         <discardingThreshold>0</discardingThreshold>
30         <queueSize>1024</queueSize>
31         <neverBlock>true</neverBlock>
```



```
26     <appender-ref ref="CONSOLE_SKY" />
27 </appender>
28
29 <root level="INFO">
30     <appender-ref ref="ASYNC" />
31     <appender-ref ref="grpc-log" />
32 </root>
33 </configuration>
```

java >

- 启动boot服务后，请求一个接口，再次重新访问http://localhost:8088打开skywalking webapp ui界面，可以看到Log中有日志数据了。

### (6-3) 接口调用链路额外信息（包括具体的方法名，参数，返回值）的监控

- 需要引入skywalking的链路追踪jar包，方法上打上@Trace、@Tag、@Tags，这些注解会被能Agent拦截到

```
1 // https://mvnrepository.com/artifact/org.apache.skywalking/apm-toolkit-trace
2 implementation("org.apache.skywalking:apm-toolkit-trace")
```

复制代码

swift >

- 使用@Trace注解标注到业务方法上，此方法就会被监控
- 使用@Tags或Tag标注到业务方法上，此方法的额外信息会被监控，比如方法参数和返回值。
- 方法参数不是对象类型



```
@Trace
@Tag(key = "body", value = "arg[0]") //有参数
@Tag(key = "StageStatisticsResponse", value = "returnedObj")//有返回值
@Tags(//有参数和有返回值
    value = [
        Tag(key = "body", value = "arg[0]"),
        Tag(key = "StageStatisticsResponse", value = "returnedObj")
    ]
)

override fun getAcuteCoronaryDiseaseStageStatistics(body: BigInteger): StageStatisticsResponse
```

- 方法参数是对象类型
  - 在对象的属性上分别添加@Tag注解，当调用业务方法时，SkyWalking APM Agent 会自动捕获对象中标记的字段信息，并将其作为追踪数据的一部分发送到 SkyWalking oap服务器。

## (7) Alerting告警和回调通知

- 具体可参考官网地址：<https://skywalking.apache.org/docs/main/v9.4.0/en/setup/backend/backend-alarm/>



2

Projects and Docs

Events

A demo music application to showcase features of Apache SkyWalking in action.

🌟 Star 74

docs ▾

🌟 Star 77

## Server and UI



### SkyWalking

SkyWalking OAP server and UI. Introduce concepts, designs, setup, and corresponding APIs referred to telemetry data report and query.

🌟 Star 21,909

docs ▾

3

Next

v9.4.0

v9.3.0

v9.2.0

v9.1.0

v9.0.0

## Agent



### Java Agent

The Java Agent for Apache SkyWalking,



### Booster UI

3rd generation UI for S

🌟 Star 119



### Python Agent

The Python Agent for /

## skywalking

SkyWalking OAP server and UI. Introduce concepts, designs, setup, and corresponding APIs referred to telemetry data report and query.

Version: v9.4.0

Welcome

Concepts and Designs

Setup

4

Quick Start

Configuration Vocabulary

Advanced Setup

Tracing

Metrics

5

OAL Scripts

OpenTelemetry Metrics

AWS CloudWatch Metrics

Zabbix Metrics

Meter Analysis

Telegraf Metrics

Apdex Threshold

# Alerting

Alerting mechanism measures system performance according to the metrics of service.

The alerting core is driven by a collection of rules defined in `config/alarm-setting`:

1. [alerting rules](#). They define how metrics alerting should be triggered and what actions to take.
2. [Webhooks](#). The list of web service endpoints, which should be called after an alert is triggered.
3. [gRPCHook](#). The host and port of the remote gRPC method, which should be called after an alert is triggered.

## Entity name

Defines the relation between scope and entity name.

- **Service**: Service name
- **Instance**: {Instance name} of {Service name}
- **Endpoint**: {Endpoint name} in {Service name}
- **Database**: Database service name
- **Service Relation**: {Source service name} to {Dest service name}
- **Instance Relation**: {Source instance name} of {Source service name} to {Dest instance name}
- **Endpoint Relation**: {Source endpoint name} in {Source Service name} to {Dest endpoint name}

## Rules

There are two types of rules: individual rules and composite rules. A composite rule is a collection of individual rules.

## Individual rules

An alerting rule is made up of the following elements:

- **Rule name.** A unique name shown in the alarm message. It must end with `_r`.
- **Metrics name.** This is also the metrics name in the OAL script. Only long, dou



## ( 7-1 ) alerting rules

- 告警规则定义在config/alarm-settings.yml文件中，告警规则中会使用度量指标来作为条件和阈值，当告警规则超过阈值后，就会回调Webhooks/gRPCHook接口（请求参数是List<org.apache.skywalking.oap.server.core.alarm.AlarmMessage>）进行消息通知。
- 度量指标是定义在OAL scripts（config/oal/\*.oal文件中）and MAL scripts,and Event中，度量指标可以分析和查询服务，实例，端点的数据。
- 总结：config/alarm-settings.yml中的告警规则配置中使用 OAL 脚本来定义条件和阈值，以触发告警。此外，你还可以使用 OAL 脚本进行自定义的数据分析和查询。

## ( 7-2 ) Webhooks（业务服务中来定义被回调接口方法，方法参数内容是AlarmMessage的字段，接口内调用三方API（微信钉钉飞书）来发消息）

- 这段话就表明了触发alerting-settings.yml中的告警规则后，skywalking将通过HTTP的POST进行异步请求一个web服务接口，以application/json的内容类型发送，JSON内容是List<org.apache.skywalking.oap.server.core.alarm.AlarmMessage>，下面是简单告警消息示例

The Webhook requires the peer to be a web container. The alarm message will be sent through HTTP post by application/json content type. The JSON format is based on List<org.apache.skywalking.oap.server.core.alarm.AlarmMessage> with the following key information

复制代码

```
1 [{
2   "scopeId": 1,
3   "scope": "SERVICE",
4   "name": "serviceA",
5   "id0": "12",
6   "id1": "",
```

```
7   "ruleName": "service_resp_time_rule",
8   "alarmMessage": "alarmMessage xxxx",
9   "startTime": 1560524171000,
10  "tags": [{
11      "key": "level",
12      "value": "WARNING"
13  }]
14 }, {
15     "scopeId": 1,
16     "scope": "SERVICE",
17     "name": "serviceB",
18     "id0": "23",
19     "id1": "",
20     "ruleName": "service_resp_time_rule",
21     "alarmMessage": "alarmMessage yyy",
22     "startTime": 1560524171000,
23     "tags": [{
24         "key": "level",
25         "value": "CRITICAL"
26     }]
27 }]
```

java >

- 当触发告警规则时，会回调url中的端点（资源接口路径）

```
作文件.txt x 每日任务记录.txt x 暂时记录.txt x alarm-settings.yml x core.oal x application.yml x application.y
1
2 endpoint_abnormal_rule:
3   metrics-name: endpoint_abnormal
4   threshold: 1
5   op: ">="
6   period: 2
7   count: 1
8   message: 接口: ${name}\n 指标: 接口异常\n 详情: 最近2分钟内至少1次 \n
9   tags:
10    level: ERROR
11
12
13 webhooks:
14   - url: http://serviceName/endpoint
15
```

- 业务服务中定义被回调的接口

复制代码

```
1 @RestController
2 @RequestMapping("/endpoint")
3 public class WebhookController {
4
5     @PostMapping
6     public ResponseEntity<String> handleWebhook(@RequestBody List<AlarmMessage>
7     alarmMessages) {
8         // 解析接收到的告警消息
9         for (AlarmMessage alarmMessage : alarmMessages) {
10             // 提取告警信息中的关键数据
11             String scope = alarmMessage.getScope();
12             String name = alarmMessage.getName();
13             // 进行业务逻辑处理 (调用微信/飞书/钉钉通知)
14             // ...
15         }
16     }
17 }
```

```

14     }
15     // 返回响应
16     return ResponseEntity.ok("Webhook received successfully");
17 }
18 }

```

java >

### ( 6-3 ) Feishu Hook ( 飞书群中使用自定义机器人来定义被回调的接口，发消息到群中 )

- 触发alerting-settings.yml中的告警规则后，skywalking会回调指定的飞书webhook地址（飞书群中先创建的自定义机器人的webhook，然后这里才能指定的回调地址），HTTP POST请求以application/json内容类型发送告警消息到飞书群中，下面是简单告警消息示例

复制代码

```

1 feishuHooks:
2   textTemplate: |-
3     {
4       //告警消息类型
5       "msg_type": "text",
6       "content": {
7         // 告警消息内容，%s是占位符，会替换为alert-settings.yml文件中告警规则的message消息
8         "text": "Apache SkyWalking Alarm: \n %s."
9       },
10      //可以配置只发送给群中的某些人，也可以不配置，直接发送到群中
11      "ats": "feishu_user_id_1,feishu_user_id_2"
12    }
13 webhooks:
14   //飞书群中自定义机器人的webhook地址
15   - url: https://open.feishu.cn/open-apis/bot/v2/hook/7a51e902-8066-4ce9-937b-e37582d71092
16   //飞书群中自定义机器人的secret密钥
17   secret: rKmxgwCmGJbTDTygJnuTKg

```



```
71 endpoint_abnormal_rule:
72   metrics-name: endpoint_abnormal
73   threshold: 1
74   op: ">="
75   period: 2
76   count: 1
77   message: 接口: ${name}\n 指标: 接口异常\n 详情: 最近2分钟内至少1次 \n
78   tags:
79     level: ERROR
80
81 feishuHooks:
82   textTemplate: |-
83     {
84       "msg_type": "text",
85       "content": {
86         "text": "Apache SkyWalking Alarm: \n %s."
87       },
88       "ats": "feishu_user_id_1,feishu_user_id_2" #可以配置只发送给群中的某些人，也可以不配置，直接发送到群中
89     }
90   webhooks:
91     - url: https://open.feishu.cn/open-apis/bot/v2/hook/7a51e902-8066-4ce9-937b-e37582d71092
92       secret: rKmxgwCmGJbTDTygJnuTKg
```

- 飞书群中创建自定义机器人并配置告警通知：见笔记2
- 重启boot应用，重启bin/startup.bat（oap服务，webapp ui服务），然后调用一个业务接口（模拟异常），看skywalking webapp ui告警面板是否能正常显示告警信息，飞书群中是否收到告警通知



