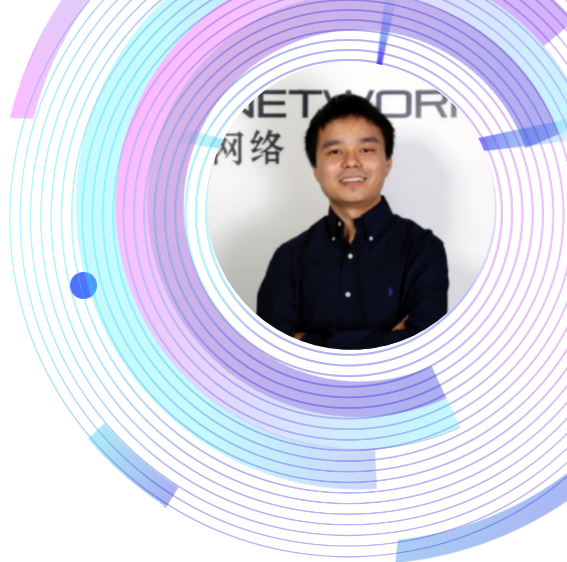


# 开启高度自动化的 可观测性新时代

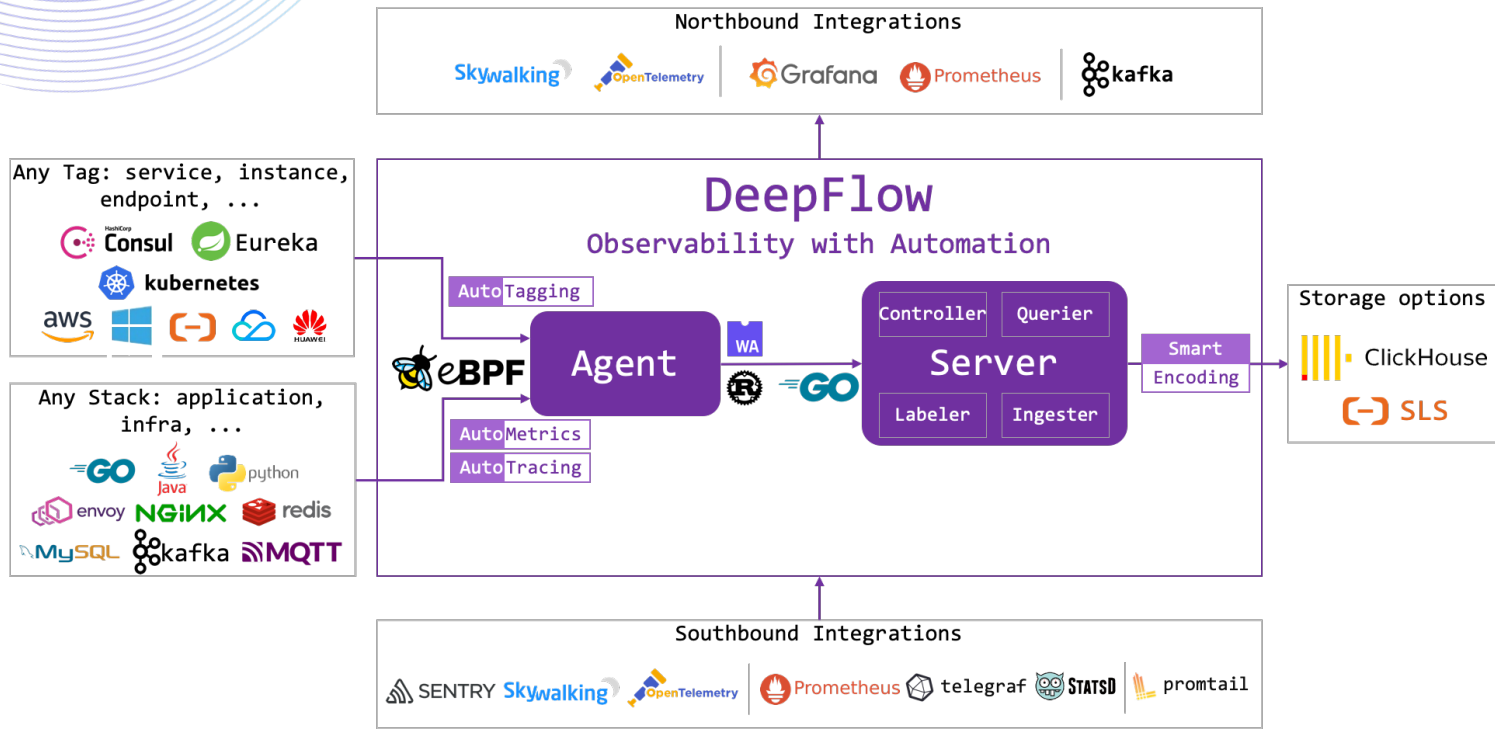
DeepFlow 首个开源版本发布

●-----> 向阳 / 云杉网络 研发VP

2022年7月27日



# DeepFlow 首个开源版本发布了



# 体验高度自动化的可观测性

DeepFlow<sup>®</sup>

AutoMetrics: 自动化的全栈性能指标、全景服务依赖

Integration: 自动化的 Prometheus、Telegraf 集成

AutoTracing: 自动化的分布式调用链追踪

Integration: 自动化的 OpenTelemetry、SkyWalking 集成



~~数据孤岛  
高基丢失  
追踪不全  
数据采样~~

# 内容目录

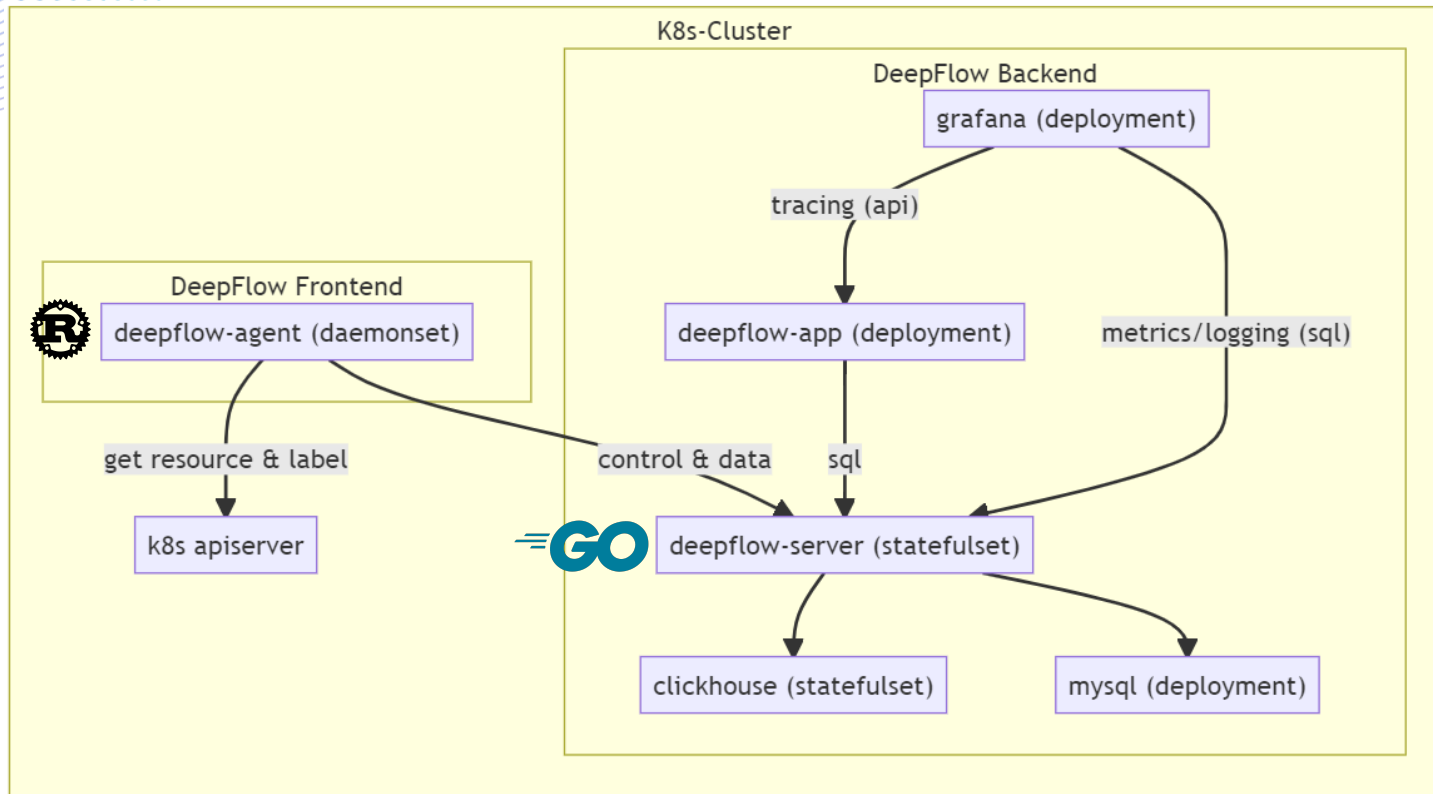
1 AutoMetrics: 自动化的全栈性能指标、全景访问关系

2 Integration: 自动化的 Prometheus、Telegraf 集成

3 AutoTracing: 自动化的分布式调用链追踪

4 Integration: 自动化的 OpenTelemetry、SkyWalking 集成

# 准备开始了





# 一步完成!

```
helm repo add deepflow https://deepflowys.github.io/deepflow
```

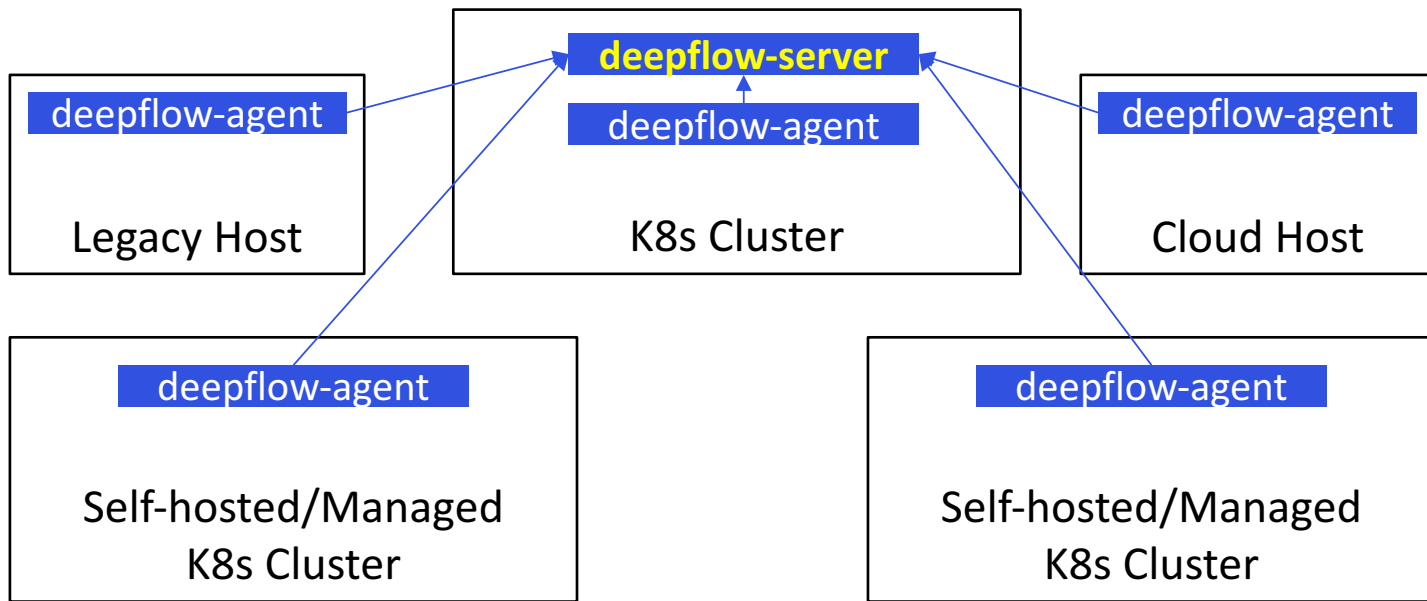
```
helm repo update deepflow
```

```
helm install deepflow -n deepflow deepflow/deepflow --create-namespace
```

<https://deepflow.yunshan.net/docs/zh/install/single-k8s/#部署-deepflow>

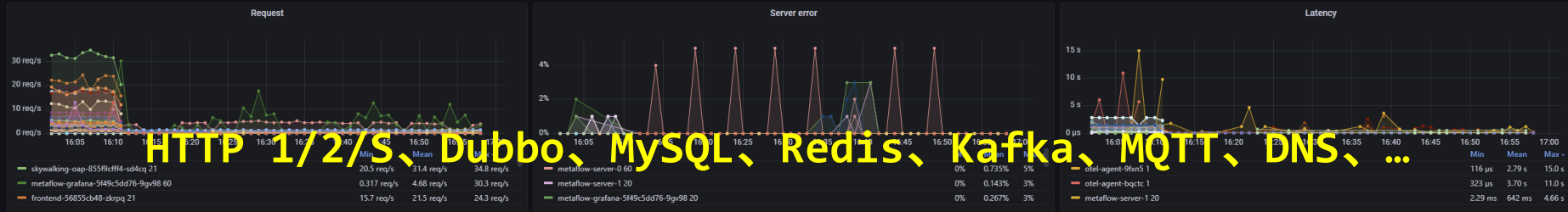
# 当然更复杂也没问题，零依赖水平扩展

- All-in-One 快速部署
- 监控单个 K8s 集群
- 监控多个 K8s 集群
- 监控传统服务器
- 监控云服务器
- 监控托管 K8s 集群



<https://deepflow.yunshan.net/docs/zh/install/overview/>

# 任意微服务的应用性能 RED 指标



HTTP 1/2/S, Dubbo, MySQL, Redis, Kafka, MQTT, DNS, ...

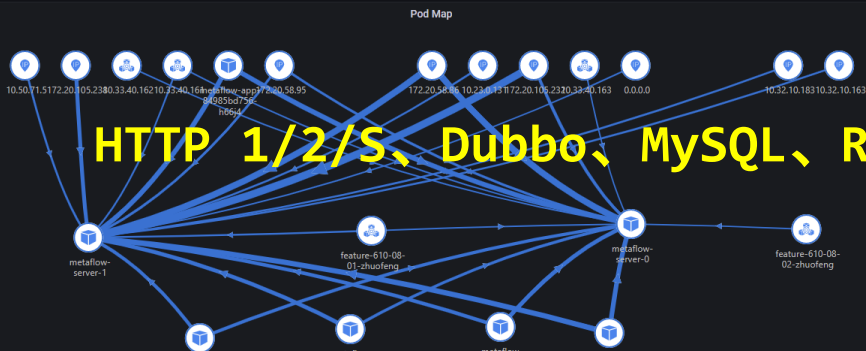
自动注入的资源、服务、K8s Label, 随心过滤, 随心聚合

零插码, 释放开发生产力

Pod name	Protocol	Request	Client error	Server error	Latency
otel-agent-bqctc	Rest	0 req/s	--	--	3.70 s
svc-item-57dd56b54b-fhbp9	HTTP	40.5 req/s	0%	0%	2.52 s
svc-item-76b55d95b7-28kc2	HTTP	40.9 req/s	0%	0%	1.27 s
nacos-0	HTTP	506 req/s	0%	0%	1.14 s
nacos-0	HTTP	503 req/s	0%	0%	1.13 s
svc-order-8696d85f9b-5d75q	HTTP	124 req/s	0%	0%	826 ms
prometheus-kube-state-metrics-748fc7164-75sx9	Rest	0 req/s	--	--	668 ms
metaflow-app-84985bd756-4z9yb	HTTP	16 req/s	0%	0%	536 ms
metaflow-grafana-5f49c5dd76-9gy98	HTTP	12.2 req/s	0.317%	8.10%	475 ms
svc-order-d87fdd71-nk8c4	HTTP	180 req/s	0%	0%	365 ms
recommendation-deployment-5db94c85cd-qlnw6	HTTP2	355 req/s	0%	0%	325 ms
web-shop-6f97c6b768-2g9cz	HTTP	291 req/s	0%	0%	231 ms
metaflow-server-0	HTTP	8.76 req/s	0%	2.23%	167 ms
metaflow-clickhouse-0	Rest	2 req/s	--	--	124 ms
otel-agent-9fkn5	HTTP2	99 req/s	0%	0%	71.5 ms
metaflow-server-0	HTTP2	10 req/s	0%	0%	20.8 ms
metaflow-grafana-5f49c5dd76-9gy98	Rest	0.0351 req/s	--	--	16.5 ms
metaflow-server-0	Rest	0.216 req/s	--	--	15.8 ms
frontend-5685cb48-zkrpq	HTTP	128 req/s	0%	0%	15.1 ms
frank-cu3-779c-965c-179m	HTTP	75.6 req/s	0%	0%	6.70 ms

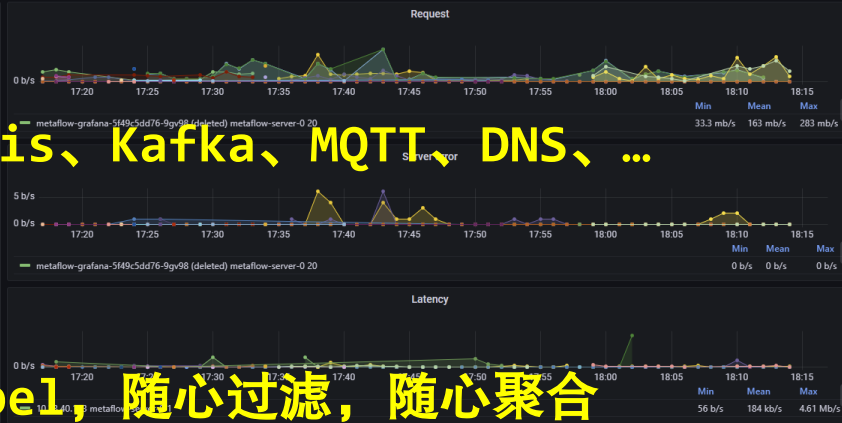


# 任意微服务的应用调用关系



HTTP 1/2/S、Dubbo、MySQL、Redis、Kafka、MQTT、Server、DNS、...

自动注入的资源、服务、K8s Label, 随心过滤, 随心聚合



Client	Server	Protocol	Request	Client error	Server error	Latency
--	metaflow-server-1	HTTP	0 req/s	0%	0%	1.77 s
--	metaflow-server-1	HTTP	0.0333 req/s	0%	0%	716 ms
--	metaflow-server-1	HTTP	0.367 req/s	0%	0%	498 ms
--	metaflow-server-0	HTTP	0.100 req/s	0%	0%	401 ms
metaflow-grafana-5149c5dd76-j47hz (deleted)	metaflow-server-0	HTTP	0.0898 req/s	0%	12.0%	390 ms
10.33.40.161	metaflow-server-0	Rest	0 req/s	--	--	326 ms
metaflow-grafana-5149c5dd76-j47hz	metaflow-server-1	HTTP	0 req/s	0%	0%	333 ms

零插码, 释放开发生产力

# 任意微服务的网络性能指标

3



Pod name	Throughput(bps)	Throughput(pps)	TCP new conn.	TCP retrans rate	TCP conn. establishment fail rate ↓	TCP conn. establishment latency	TCP/UDP data latency
discounts-v1-5d9dd845cc-j8vrl	59.3 kb/s	18.5 p/s	1.18 c/s	0%	0%	56.6 μs	647 μs
loki-0	21.3 kb/s	6.31 p/s	0.77 c/s	0%	0%	40.6 μs	1.97 ms
travels-v3-7755c965c5-k7rlp	30.3 kb/s	8.71 p/s	0.646 c/s	0%	6.89%	44.5 μs	5.22 ms
travels-v2-7ccddd4df-5bk8t	47.8 kb/s	14.1 p/s	1.10 c/s	0%	6.80%	51.8 μs	5.14 ms
prometheus-server-6bbf87b66f-f9hjd	574 kb/s	41.4 p/s	0.283 c/s	0.000696%	5.88%	44.7 μs	32.7 ms
travels-v1-c49b8dc75-tdhtr	47.3 kb/s	13.9 p/s	1.16 c/s	0%	4.91%	42.5 μs	4.96 ms
viaggi-856756965f-xd9pk	15.5 kb/s	6.75 p/s	0.602 c/s	0.0257%	4.56%	41.8 μs	5.35 ms
voyages-779fb6ddd4-x5sv2	15.5 kb/s	6.72 p/s	0.598 c/s	0.0155%	3.51%	41.0 μs	3.61 ms
travels-7c775f57cf-k6pjs	15.4 kb/s	6.75 p/s	0.598 c/s	0.0134%	3.38%	44.8 μs	3.94 ms
flights-v1-657dc98844-kw4wd	23.5 kb/s	13.1 p/s	1.78 c/s	0%	2.79%	155 μs	3.60 ms
insurances-v1-79c476bf9-jc87	52.4 kb/s	12.7 p/s	1.27 c/s	0%	2.68%	51.2 μs	2.88 ms
cars-v1-76dfbcf4b4-gtxs2	23.0 kb/s	12.7 p/s	1.27 c/s	0%	2.03%	145 μs	3.36 ms
hotels-v1-5b4896ffd8-kghbz	93.3 kb/s	43.3 p/s	6.93 c/s	0%	1.80%	52.7 μs	3.00 ms
metallflow-grafana-5f49c5dd76-9gv98	641 kb/s	54.3 p/s	0.420 c/s	0.452%	0.446%	11.4 ms	49.5 ms
productpage-v1-6b746f74dc-pzmr6	14.7 kb/s	9.78 p/s	0.993 c/s	0%	0.0934%	6.08 ms	6.11 ms
details-v1-79f774bd99-pzlvw	7.73 kb/s	5.91 p/s	0.553 c/s	0%	0.0591%	49.3 μs	2.24 ms
ratings-v1-b6994bb9-vs245	5.99 kb/s	5.64 p/s	0.542 c/s	0%	0.0537%	41.1 μs	21.1 ms
reviews-v3-84779c7bbc-js9sg	7.18 kb/s	5.57 p/s	0.535 c/s	0%	0.0513%	43.9 μs	1.05 ms
metallflow-server-1	162 kb/s	37.9 p/s	0.472 c/s	0.0121%	0.00545%	2.89 ms	90.3 ms
frontend-56855c7h48-zkmm	316 kb/s	128 n/s	0.441 c/s	0.0144%	0%	—	5.00 ms

自动注入的资源、服务、K8s Label, 与应用无缝关联

零插码, 释放开发生产力

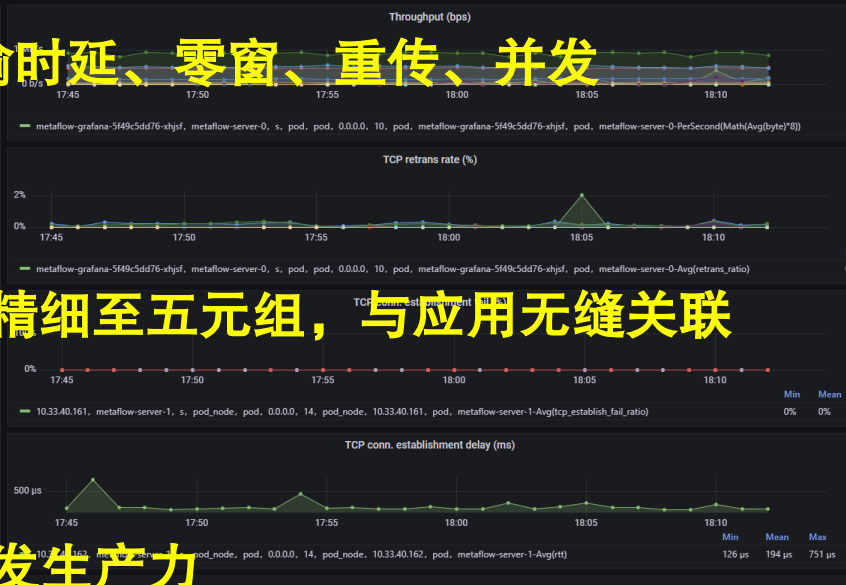
# 任意微服务的网络调用关系

4

吞吐、建连异常、建连时延、传输时延、零窗、重传、并发

自动注入的资源、服务 K8s Label, 精细至五元组, 与应用无缝关联

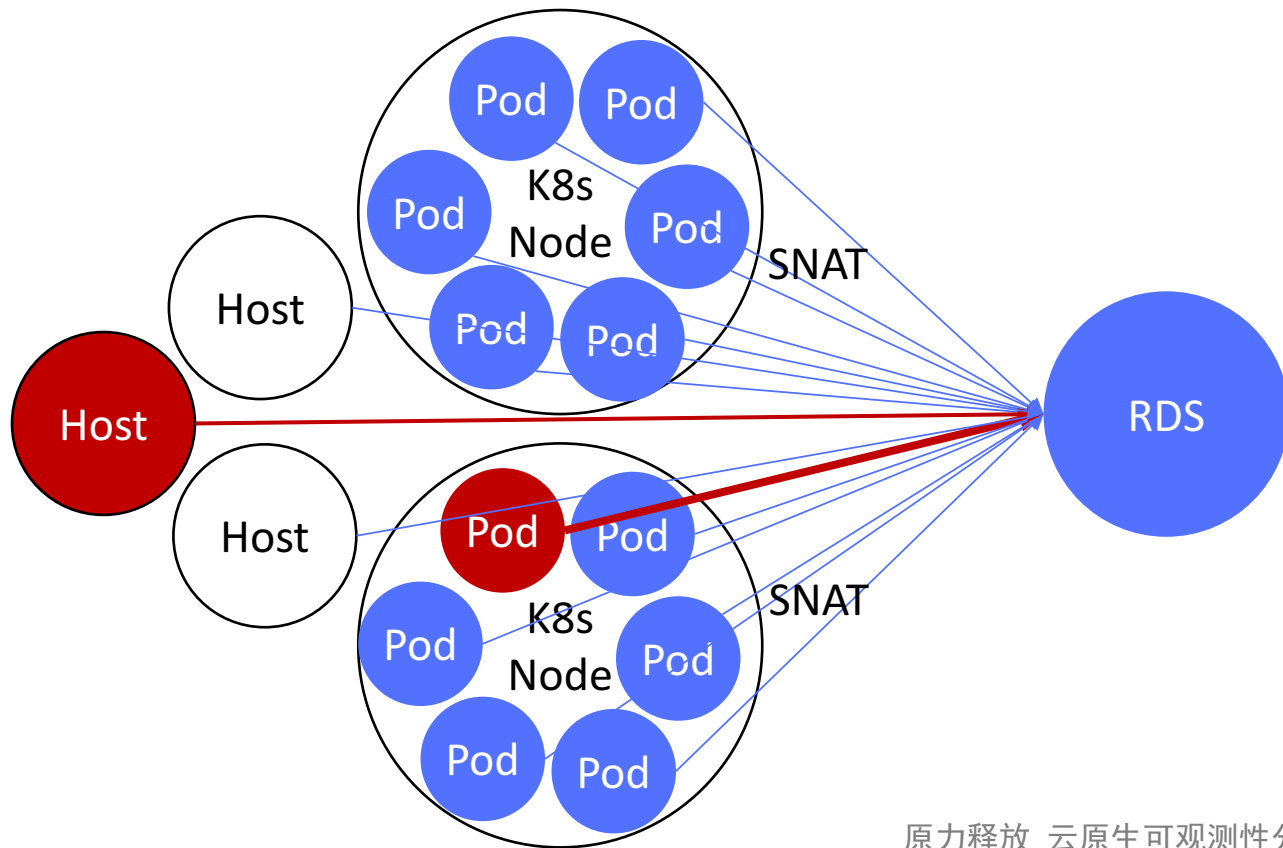
零插码, 释放开发生产力



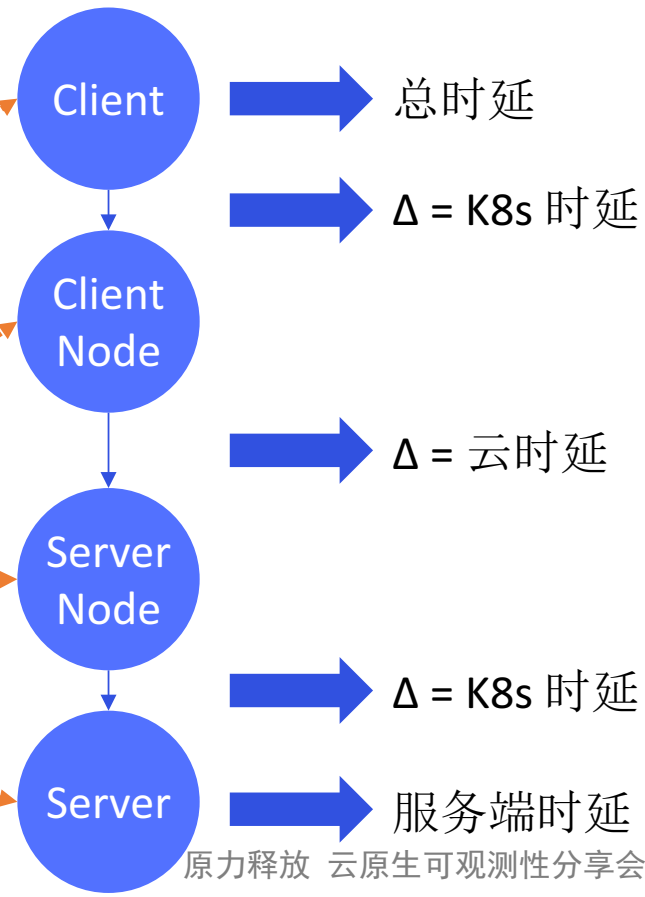
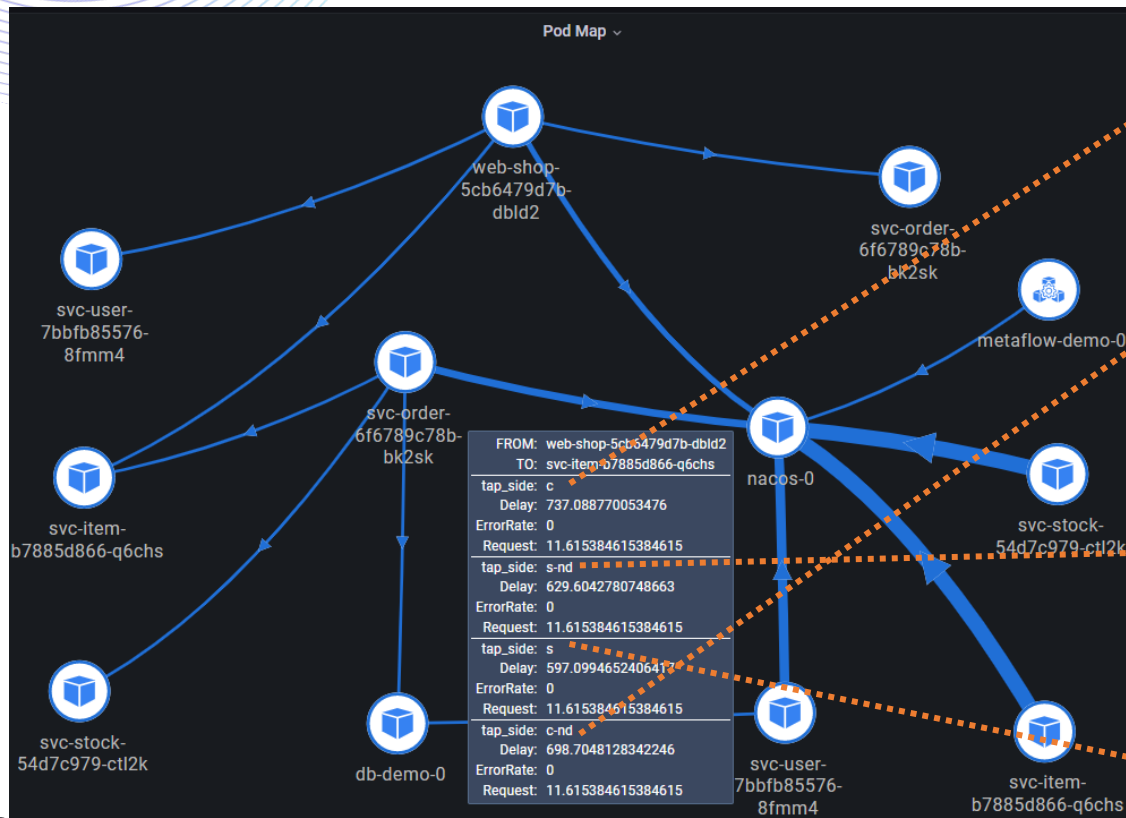
Client	Server	Tap side	Protocol	Server port	Throughput (bps)	Throughput (pps)	TCP retrans rate	TCP conn. establishn	TCP conn. establishn	TCP/UDP data delay	client_resource_type	client_resource_id	client_resource	server_resource_type	server_reso
--	metaflow-server-1	s	TCP	20416	989 b/s	0.270 p/s	0%	--	46.5 ms	4.20 ms	ip	10.50.71.5(0)	10.50.71.5	pod	
10.33.40.162	metaflow-server-1	s	TCP	20417	23.7 kb/s	44.7 p/s	0%	--	194 µs	341 µs	pod_node	1	10.33.40.162	pod	
10.33.40.161	metaflow-server-0	s	TCP	20035	78.0 kb/s	2.76 p/s	0.0387%	0%	159 µs	43.4 ms	pod_node	2	10.33.40.161	pod	
metaflow-grafana-5f49c5dd76-xhjsf	metaflow-server-1	s	TCP	20416	60.4 kb/s	1.99 p/s	0.0215%	--	153 µs	304 ms	pod	500	metaflow-grafana-5...	pod	
metaflow-app-84985bd756-h66j4	metaflow-server-0	s	TCP	20416	5.12 kb/s	2.02 p/s	0%	0%	147 µs	83.1 ms	pod	496	metaflow-app-8498...	pod	

# 全景访问关系解决什么问题 — 谁在访问 RDS

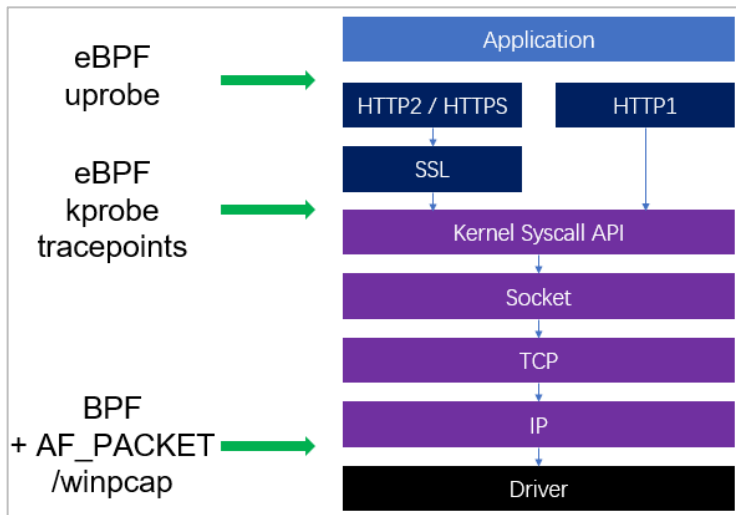
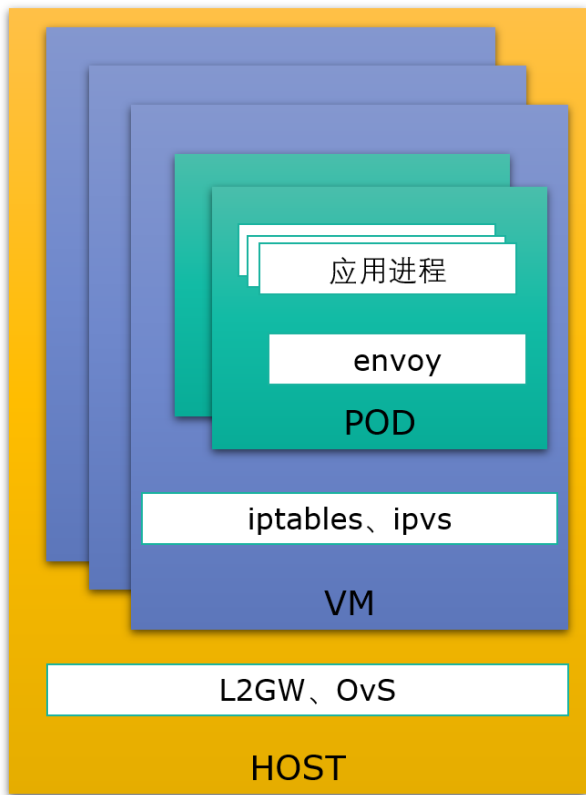
能定位到 Pod?  
能全覆盖插码?



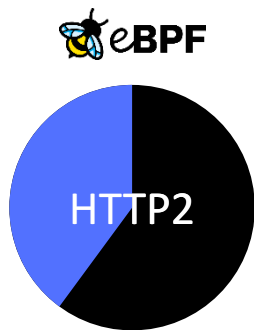
# 全栈性能指标解决什么问题 —— 故障出在哪里



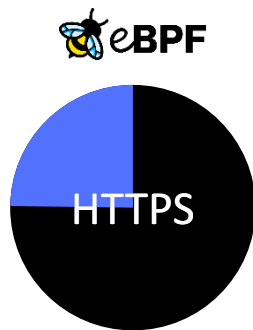
# 它是怎么做到的



# 未来迭代的方向



静态压缩表  
动态压缩表



Golang  
C/C++  
Java  
Python  
...



可编程

# 期待大家的互动

- 目前为止，你感觉 DeepFlow 怎样
- 直播间敲 **1**，意为“咦，不错哟！”
- 直播间敲 **2**，意为“啊，真棒呀！！”
- 直播间敲 **6**，意为“溜，一级棒！！！”



期待 GitHub Star 😊

[github.com/deepflowys/deepflow](https://github.com/deepflowys/deepflow)



# 内容目录

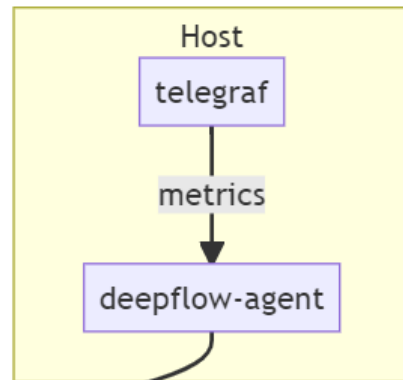
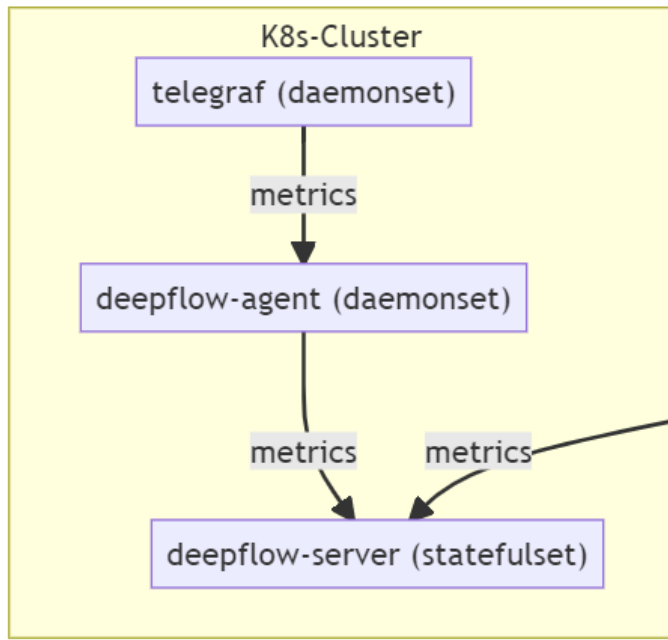
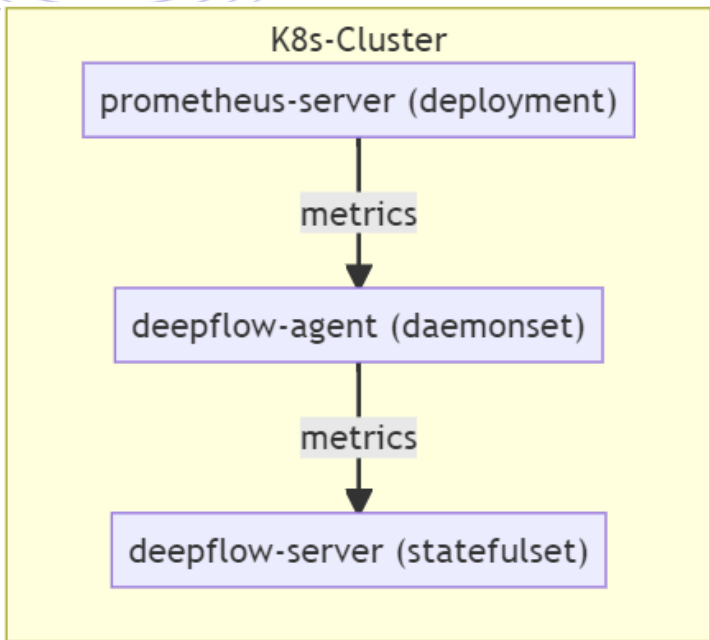
1 AutoMetrics: 自动化的全栈性能指标、全景访问关系

2 Integration: 自动化的 Prometheus、Telegraf 集成

3 AutoTracing: 自动化的分布式调用链追踪

4 Integration: 自动化的 OpenTelemetry、SkyWalking 集成

# 准备开始了



# 两步完成!

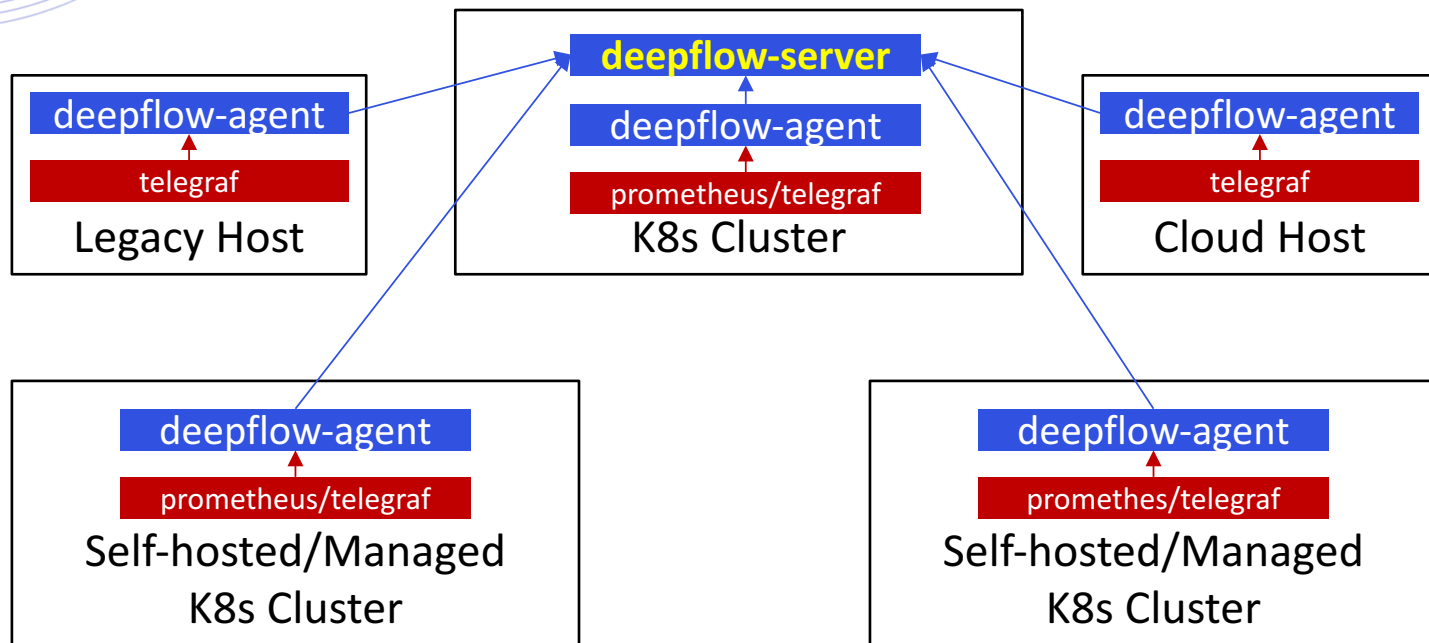
```
# prometheus-server config
remote_write:
  - url: http://${DEEPFLOW_AGENT_SVC}/api/v1/prometheus

# telegraf config
[[outputs.http]]
  url = "http://${DEEPFLOW_AGENT_SVC}/api/v1/telegraf"
  data_format = "influx"

# deepflow config
vtap_group_id: <your-agent-group-id>
external_agent_http_proxy_enabled: 1 # 默认关闭, 零端口监听
```

<https://deepflow.yunshan.net/docs/zh/agent-integration/metrics/prometheus/>

# 当然更复杂也没问题，零依赖水平扩展



# 解决什么问题 —— 下钻、关联，零插码

APP: General Metrics  
DATABASE: ext\_metrics, prometheus\_workq...  
GROUP BY: pod  
INTERVAL: pod\_cluster (K8s容器集群)  
SELECT: pod\_ns (K8s命名空间)  
WHERE: pod\_node (K8s容器节点)  
HAVING: pod\_service (K8s容器服务)  
ORDER BY: pod\_group (K8s工作负载)  
LIMIT: 100  
pod (K8s容器POD)  
resource\_gl0\_type (类型-容...)  
resource\_gl0 (资源-容器PC)

资源申请时  
定义标签

2

容器

APP: General Metrics  
DATABASE: ext\_metrics, prometheus\_workq...  
GROUP BY: label  
INTERVAL: label.version  
SELECT: label.release  
WHERE: label.k8s-app  
HAVING: label.chart  
ORDER BY: label.heritage  
LIMIT: 100  
label.istio  
label.app  
label.role

业务上线时  
定义标签

3

自定义

原始

APP: General Metrics  
DATABASE: ext\_metrics, prome...  
GROUP BY: TAG  
INTERVAL: region (区域)  
SELECT: az (可用区)  
WHERE: host (宿主机)  
HAVING: chost (云服务器)  
ORDER BY: vpc (VPC)  
LIMIT: 100  
subnet (子网)  
router (路由器)  
dhcpgw (DHCP网关)

1

资源申请时  
定义标签

APP: General Metrics  
DATABASE: ext\_metrics, prometheus\_workq...  
GROUP BY: TAG  
INTERVAL: tag.instance  
SELECT: tag.name  
WHERE: tag.job  
HAVING: tag.kubernetes\_io\_os  
ORDER BY: tag.kubernetes\_io\_arch  
LIMIT: 100  
tag.beta\_kubernetes\_io\_os  
tag.kubernetes\_io\_hostname  
tag.beta\_kubernetes\_io\_arch

4

减少在指标  
中注入标签  
释放研发生产力

# 解决什么问题 — 消除性能焦虑和高基烦恼

- ClickHouse 稀疏索引解决高基数问题

- 自动同步标签

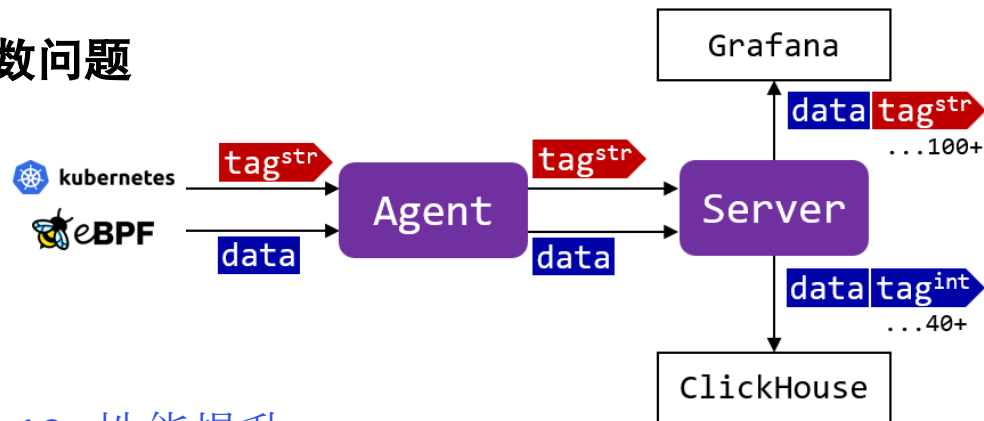
- 云资源
- K8s 资源
- K8s Label

- 自动注入编码后的标签

- 云资源
- K8s 资源

- 查询时自动关联自定义标签

- K8s Label



10x 性能提升

类型	标签字段类型	CPU 用量	内存用量	磁盘用量
基线 (SmartEncoding)	Int	1	1	1
写时编码	LowCard(string)	10	1	1.5
无编码	string	5	1.5	7.5

<https://deepflow.yunshan.net/docs/zh/auto-tagging/smart-encoding/>

# 解决什么问题 — 激活团队协同

运维和开发  
关心的

开发和运营  
关心的

自动采集

自动采集

手工插码

系统  
指标

应用  
指标

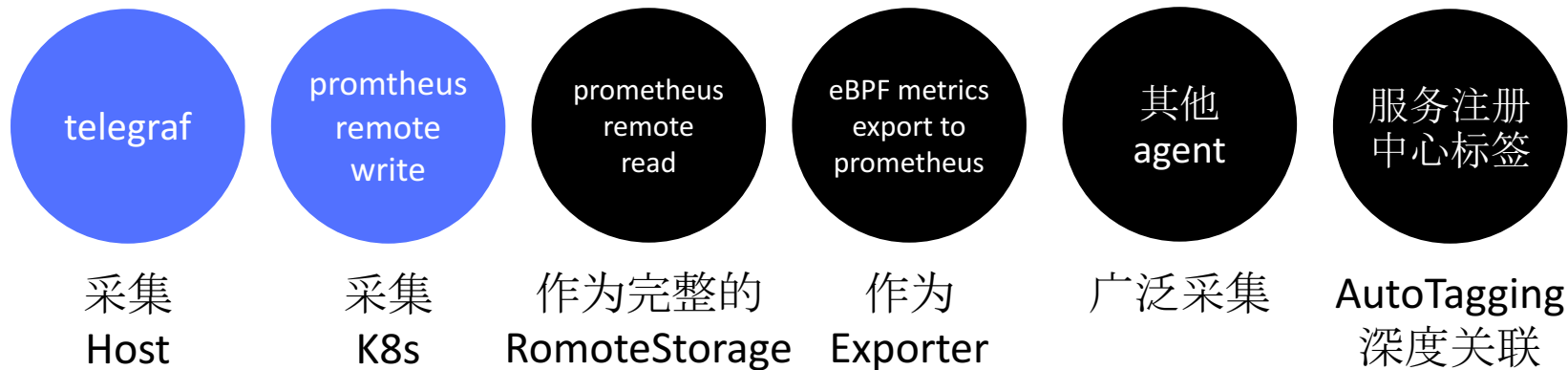
业务  
指标

DeepFlow  
Promethes  
Telegraf  
.....

DeepFlow

Exporter  
StatsD  
...

# 未来迭代的方向





# 期待大家的互动

- 目前为止，你感觉 DeepFlow 怎样
- 直播间敲 **1**，意为“咦，不错哟！”
- 直播间敲 **2**，意为“啊，真棒呀！！”
- 直播间敲 **6**，意为“溜，一级棒！！！”



期待 GitHub Star 😊

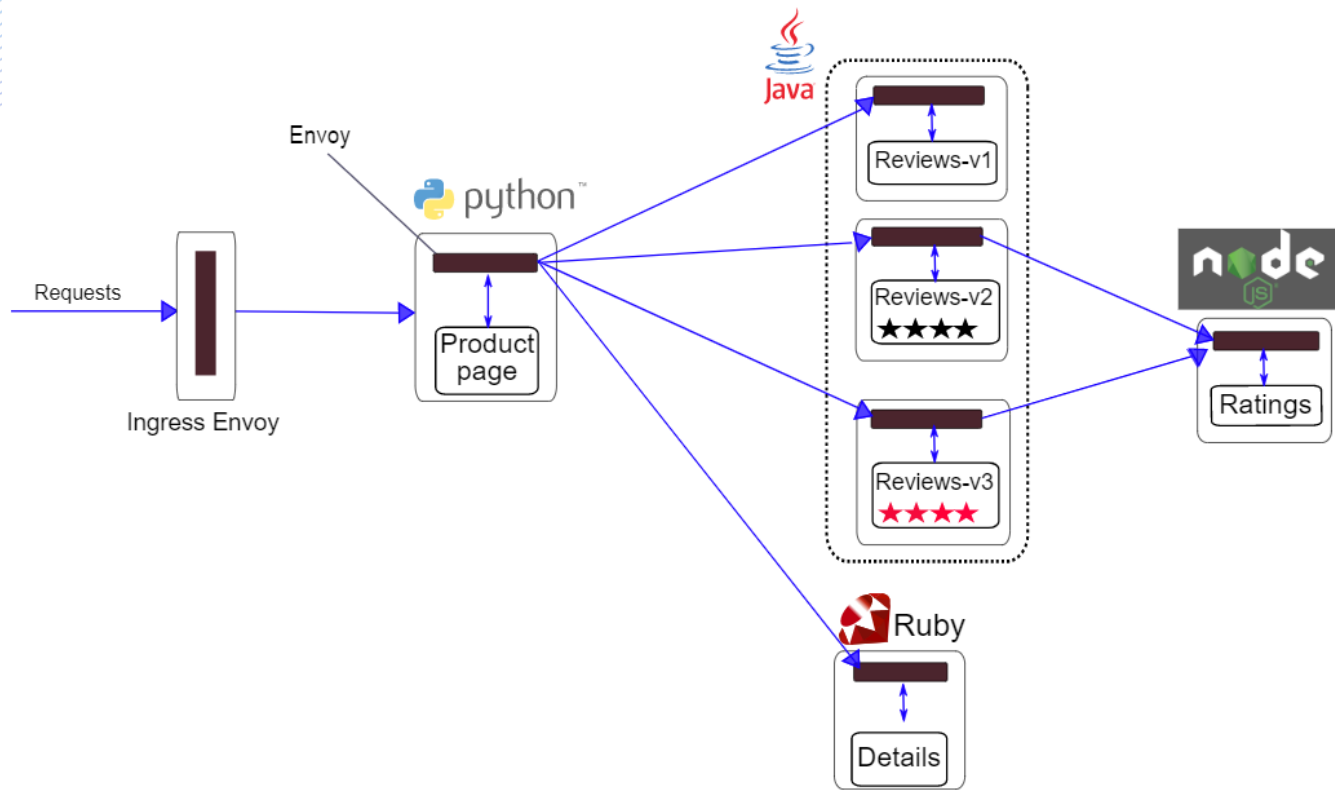
[github.com/deepflowys/deepflow](https://github.com/deepflowys/deepflow)

# 内容目录

- 1 AutoMetrics: 自动化的全栈性能指标、全景访问关系
- 2 Integration: 自动化的 Prometheus、Telegraf 集成
- 3 AutoTracing: 自动化的分布式调用链追踪**
- 4 Integration: 自动化的 OpenTelemetry、SkyWalking 集成

# 准备开始了

## Istio Bookinfo Demo





# Jaeger 追踪的怎样

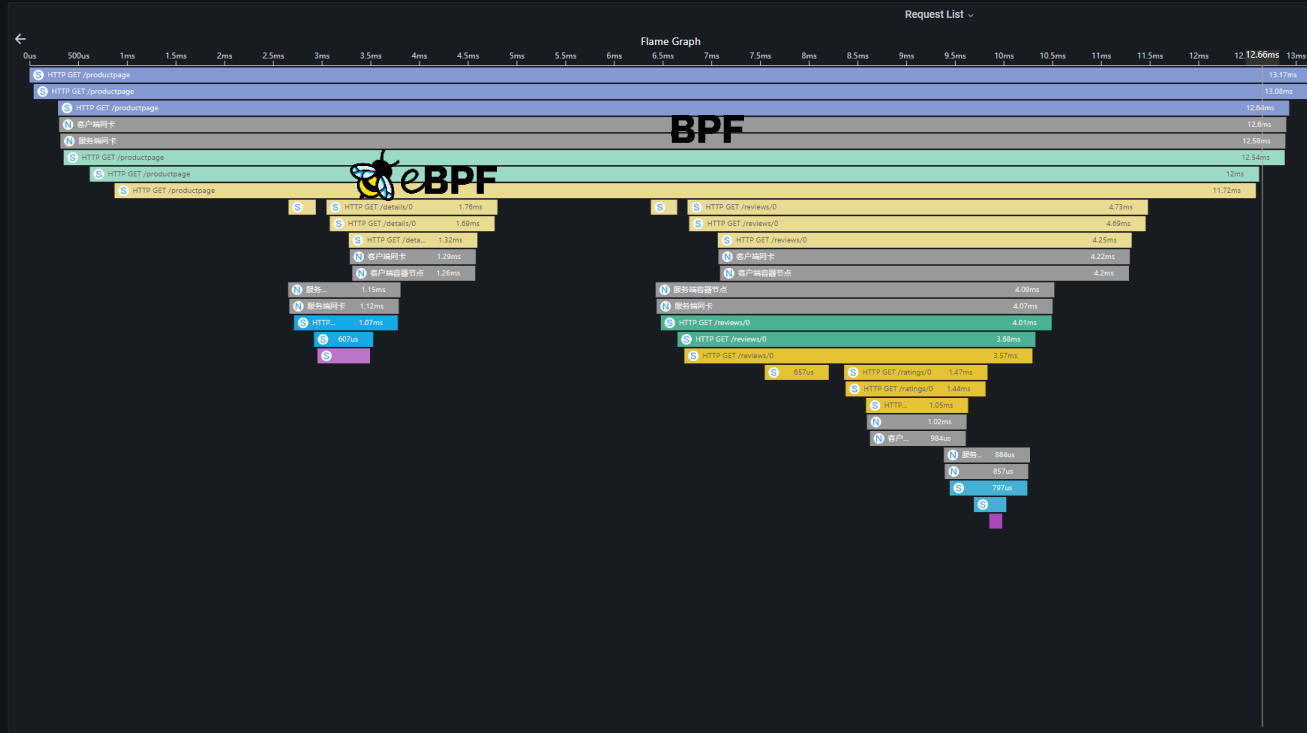
单击此处添加文本



零步完成!

# deepflow is already there :)

# DeepFlow AutoTracing, 零插码、全自动



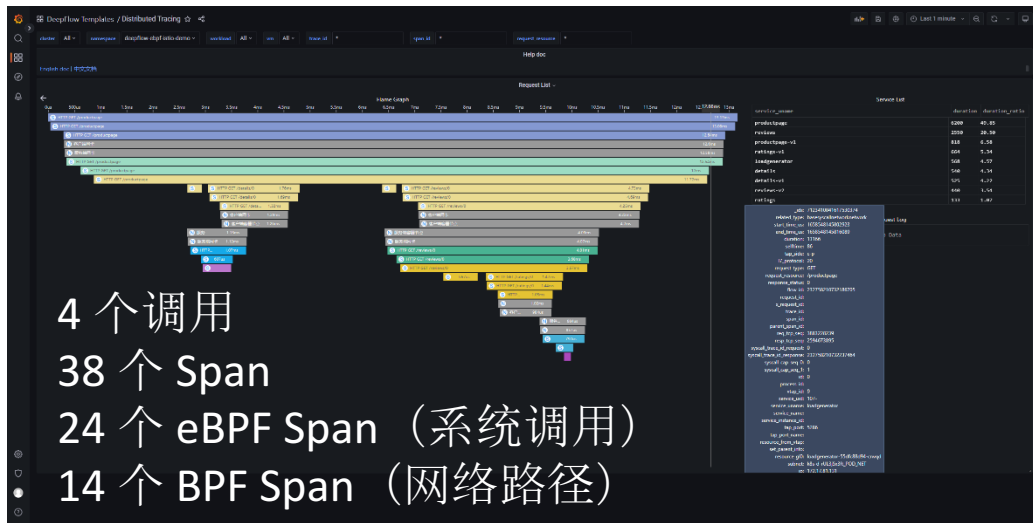
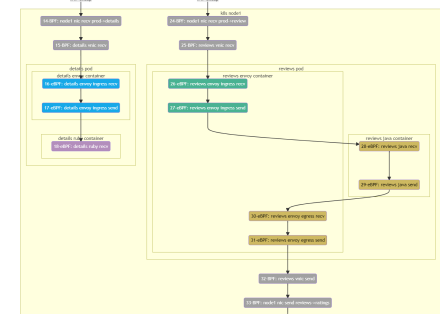
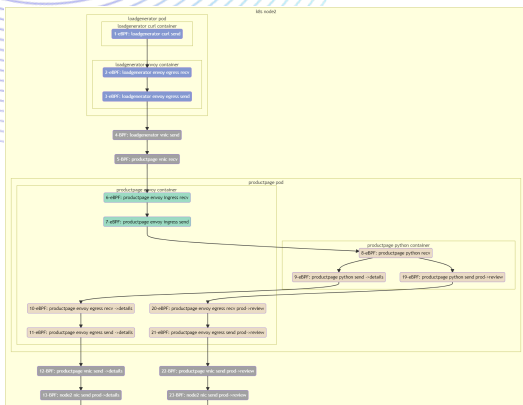
service_uname	duration	duration_ratio
productpage	6280	49.85
reviews	2550	20.58
productpage-v1	818	6.58
ratings-v1	664	5.34
loadgenerator	568	4.57
details	540	4.34
details-v1	525	4.22
reviews-v2	440	3.54
ratings	133	1.07


```
id: 7123410041617530374
related_type: basesyscallnetworknetwork
start_time_us: 1658548145802923
end_time_us: 1658548145816089
duration: 13166
selftime: 86
tap_side: c-p
i7_protocol: 20
request_type: GET
request_resource: /productpage
response_status: 0
flow_id: 232758210732186705
request_id:
x_request_id:
trace_id:
span_id:
parent_span_id:
req_top_seq: 3883228239
resp_top_seq: 2594673895
syscall_trace_id_request: 0
syscall_trace_id_response: 232758210732237464
syscall_cap_seq_0: 0
syscall_cap_seq_1: 1
id: 0
process_id:
vtap_id: 9
service_uid: 107-
service_uname: loadgenerator
service_instance_id:
tap_port: 5286
tap_port_name:
resource_from_vtap:
set_parent_info:
resource_gid: loadgenerator-55dfc88d94-crwq
subnet: k8s-d-1u13j5x3r_POD_NET
ip: 172.17.0.131
```

Request Log  
Data



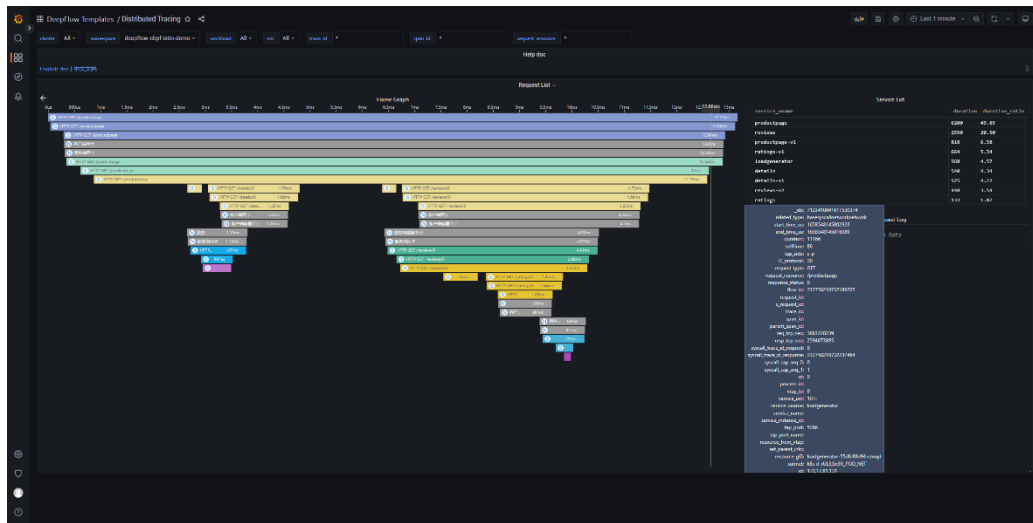
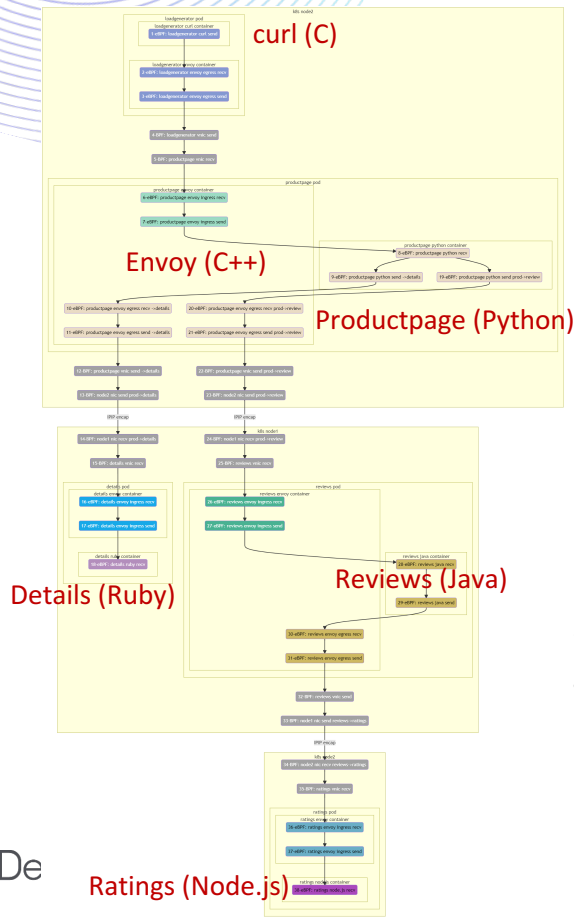
# 感受 DeepFlow 的自动化分布式追踪




1. 零插码：且无需向 HTTP 头注入 TraceID 或 SpanID  eBPF
2. 全链路：4 个调用、38 个 Span，分为 24 eBPF Span + 14 BPF Span



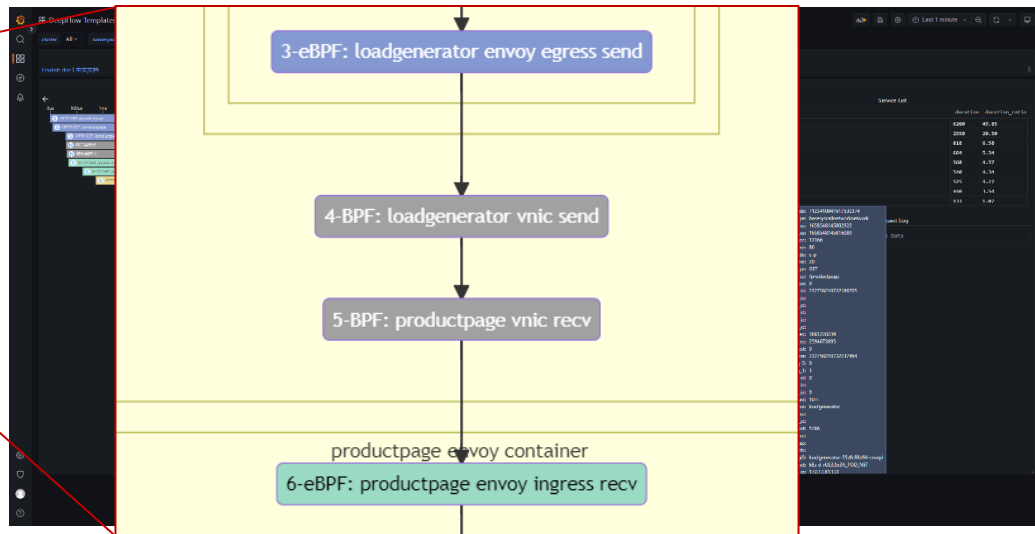
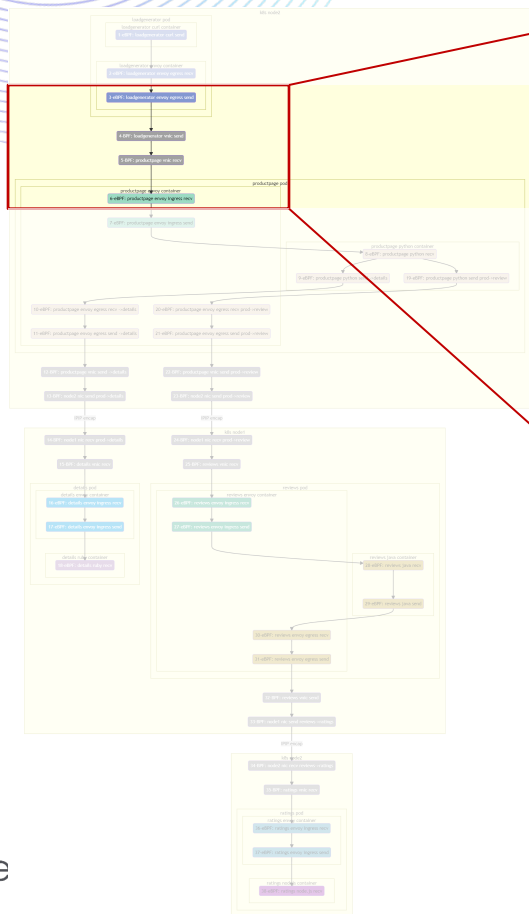
# 感受 DeepFlow 的自动化分布式追踪




3

1. 零插码：且无需向 HTTP 头注入 TraceID 或 SpanID  eBPF
2. 全链路：4 个调用、38 个 Span，分为 24 eBPF Span + 14 BPF Span
3. 多语言：Java、Python、Ruby、Node.js 及 C/C++ (curl/envoy)

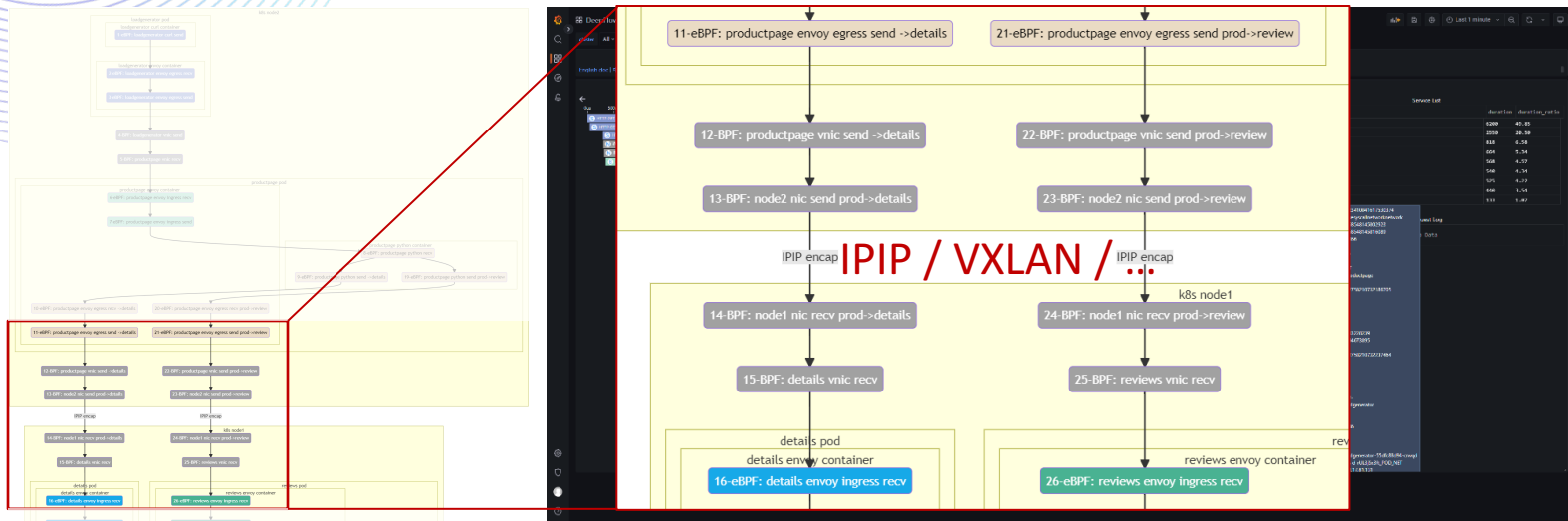
# 感受 DeepFlow 的自动化分布式追踪




4

1. 零插码：且无需向 HTTP 头注入 TraceID 或 SpanID  eBPF
2. 全链路：4 个调用、38 个 Span，分为 24 eBPF Span + 14 BPF Span
3. 多语言：Java、Python、Ruby、Node.js 及 C/C++ (curl/envoy)
4. 全栈：追踪同 K8s Node 上两个 Pod 之间的网络路径

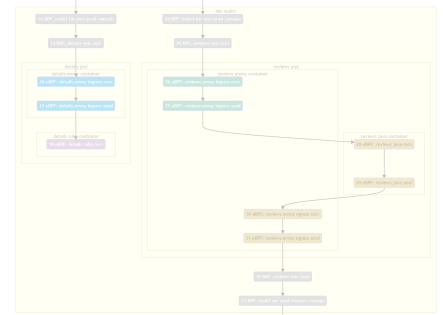
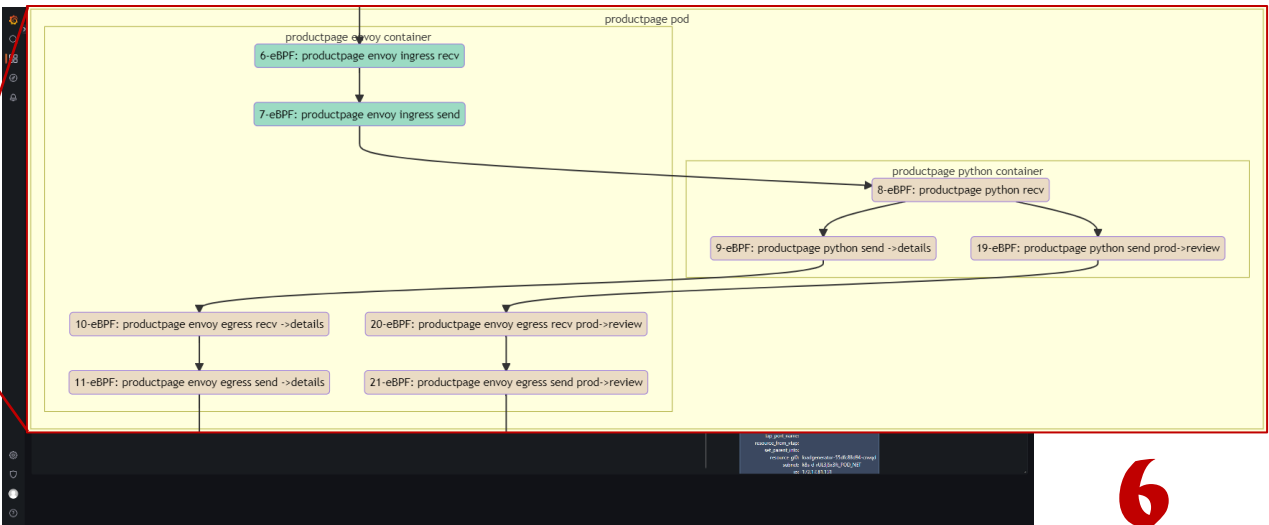
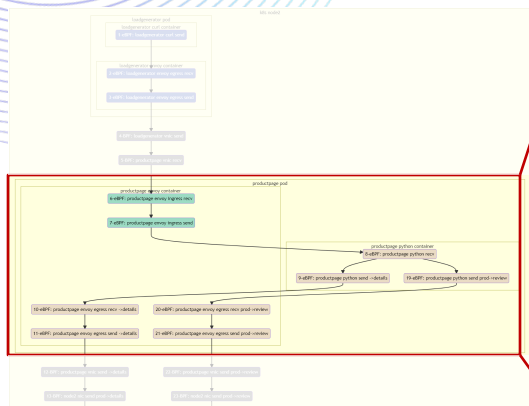
# 感受 DeepFlow 的自动化分布式追踪




5

1. 零插码：且无需向 HTTP 头注入 TraceID 或 SpanID  eBPF
2. 全链路：4 个调用、38 个 Span，分为 24 eBPF Span + 14 BPF Span
3. 多语言：Java、Python、Ruby、Node.js 及 C/C++ (curl/envoy)
4. 全栈：追踪同 K8s Node 上两个 Pod 之间的网络路径
5. 全栈：追踪跨 K8s Node 上两个 Pod 之间的网络路径，即使有隧道

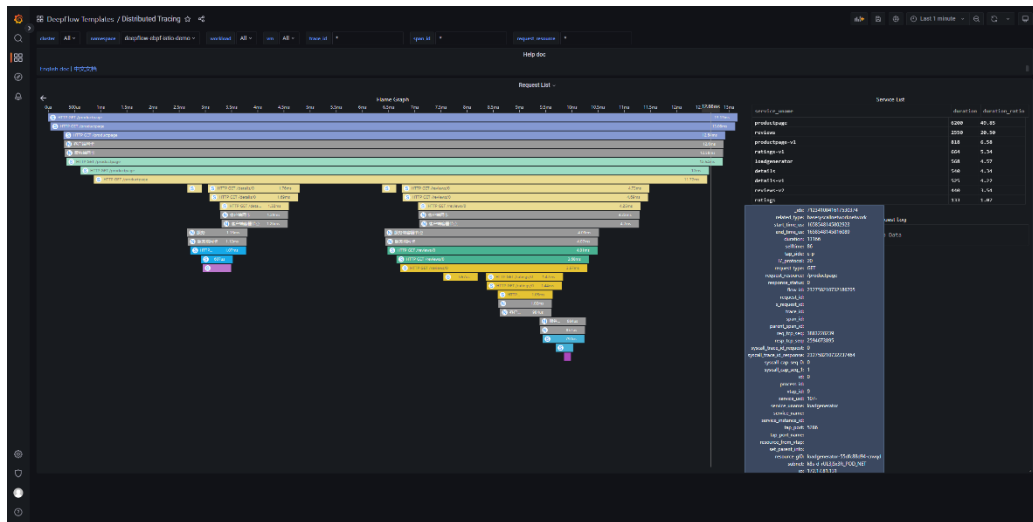
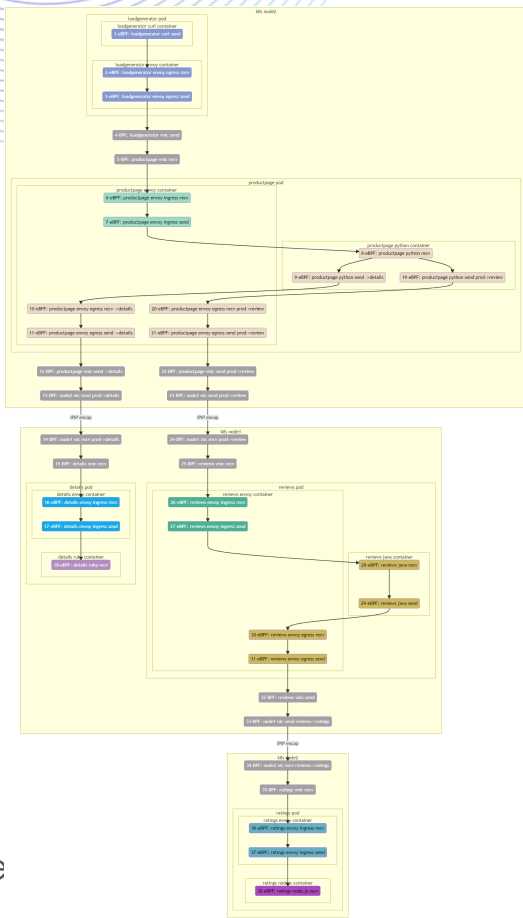
# 感受 DeepFlow 的自动化分布式追踪




6

1. 零插码：且无需向 HTTP 头注入 TraceID 或 SpanID  eBPF
  2. 全链路：4 个调用、38 个 Span，分为 24 eBPF Span + 14 BPF Span
  3. 多语言：Java、Python、Ruby、Node.js 及 C/C++ (curl/envoy)
  4. 全栈：追踪同 K8s Node 上两个 Pod 之间的网络路径
  5. 全栈：追踪跨 K8s Node 上两个 Pod 之间的网络路径，即使有隧道
  6. 全栈：追踪 Pod 内从 Envoy Ingress → 服务 → Envoy Egress 全过程
- 原力释放 云原生可观测性分享会

# 感受 DeepFlow 的自动化分布式追踪



1. 零插码：且无需向 HTTP 头注入 TraceID 或 SpanID  eBPF
2. 全链路：4 个调用、38 个 Span，分为 24 eBPF Span + 14 BPF Span
3. 多语言：Java、Python、Ruby、Node.js 及 C/C++ (curl/envoy)
4. 全栈：追踪同 K8s Node 上两个 Pod 之间的网络路径
5. 全栈：追踪跨 K8s Node 上两个 Pod 之间的网络路径，即使有隧道
6. 全栈：追踪 Pod 内从 Envoy Ingress → 服务 → Envoy Egress 全过程

<https://deepflow.yunshan.net/docs/zh/auto-tracing/istio-bookinfo-demo/>

# 未来迭代的方向



BIO

线程并发



NIO

负载均衡



AIO

协程

QCon 2022 详细拆解

<https://qcon.infoq.cn/2022/beijing/track/1293>

# 期待大家的互动

- 目前为止，你感觉 DeepFlow 怎样
- 直播间敲 **1**，意为“咦，不错哟！”
- 直播间敲 **2**，意为“啊，真棒呀！！”
- 直播间敲 **6**，意为“溜，一级棒！！！”



期待 GitHub Star 😊

[github.com/deepflowys/deepflow](https://github.com/deepflowys/deepflow)

# 内容目录

1 AutoMetrics: 自动化的全栈性能指标、全景访问关系

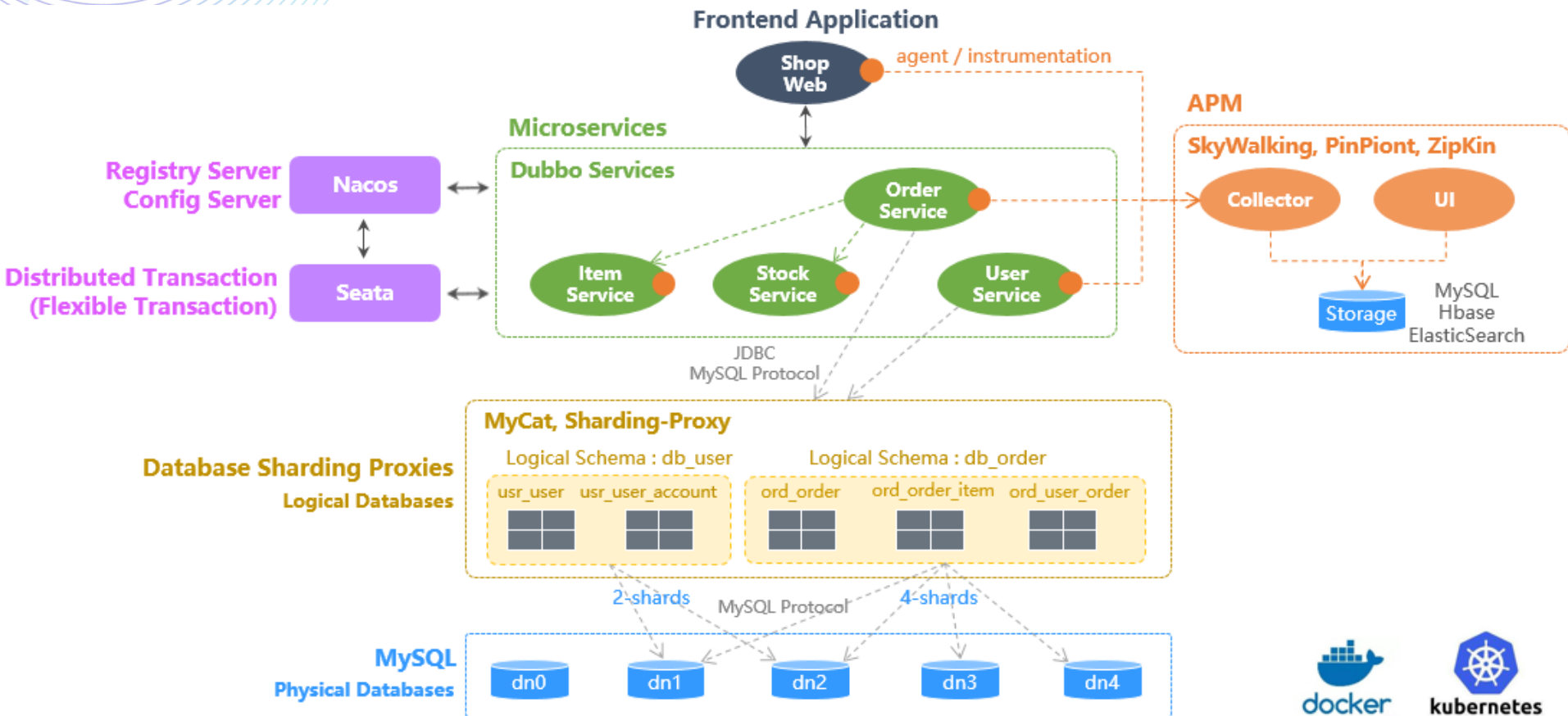
2 Integration: 自动化的 Prometheus、Telegraf 集成

3 AutoTracing: 自动化的分布式调用链追踪

**4 Integration: 自动化的 OpenTelemetry、SkyWalking 集成**



# 我们来追踪一个 Spring Boot 应用

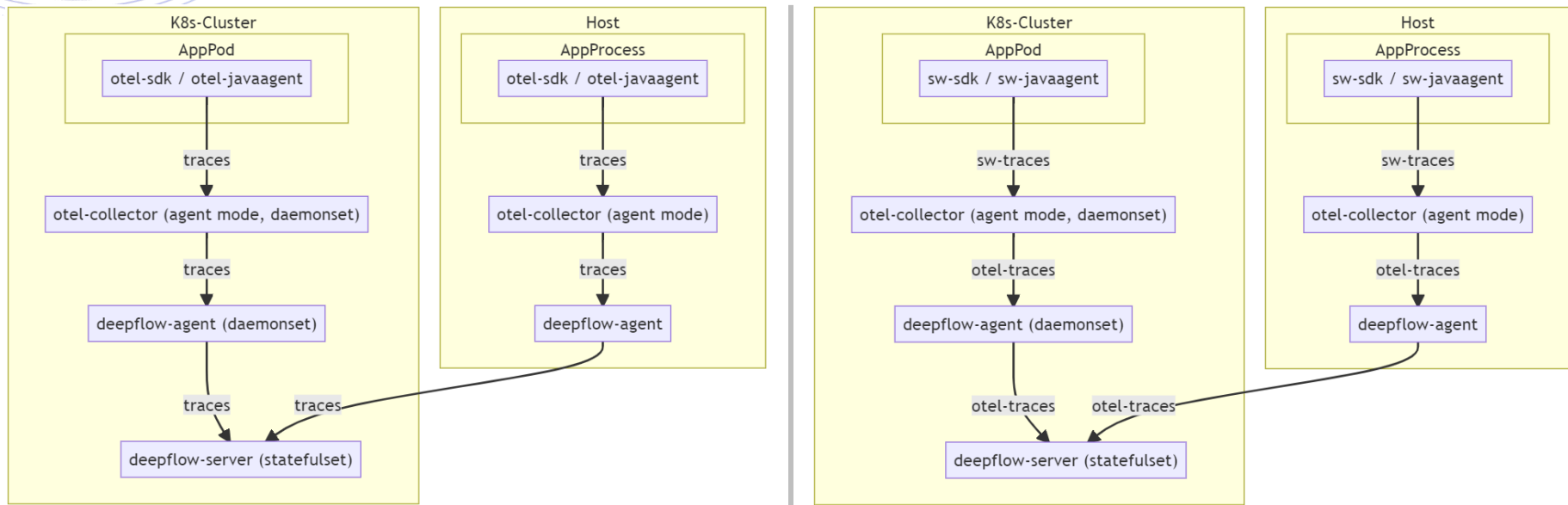


# OpenTelemetry + Jaeger 追踪结果如何



46 Spans

# DeepFlow 准备开始了



OpenTelemetry

OpenTelemetry + SkyWalking

# 两步完成！ Hello OpenTelemetry!

## # otel-agent config

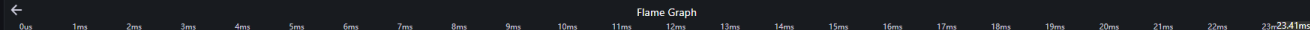
```
otlphttp:  
  traces_endpoint: "http://${HOST_IP}:38086/api/v1/otel/trace"  
  tls:  
    insecure: true  
  retry_on_failure:  
    enabled: true
```

## # deepflow config

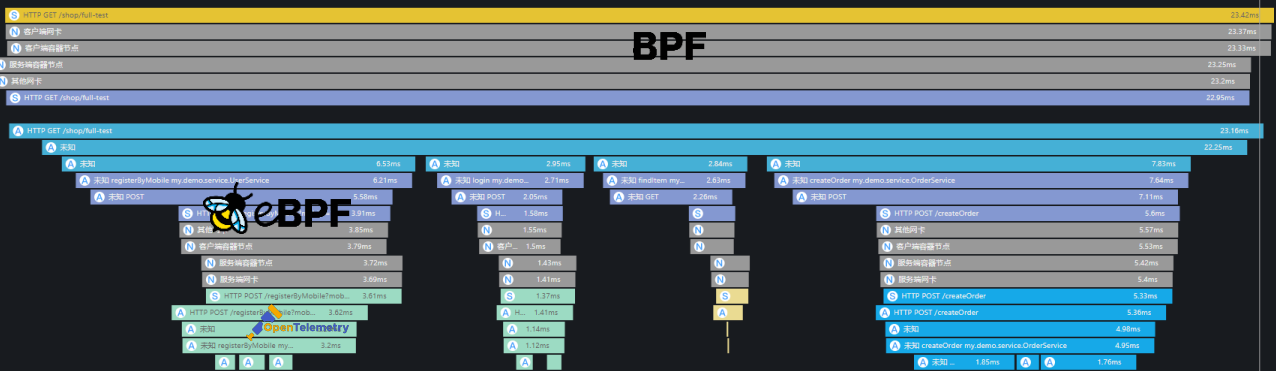
```
vtap_group_id: <your-agent-group-id>  
external_agent_http_proxy_enabled: 1 # 默认关闭，零端口监听
```

<https://deepflow.yunshan.net/docs/zh/agent-integration/tracing/opentelemetry/>

# DeepFlow 追踪完成，无盲点



BPF



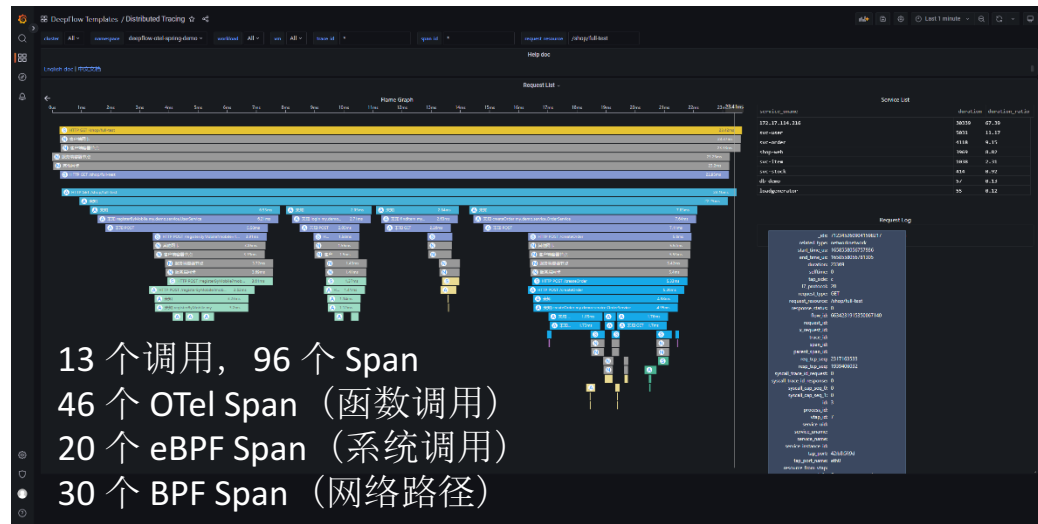
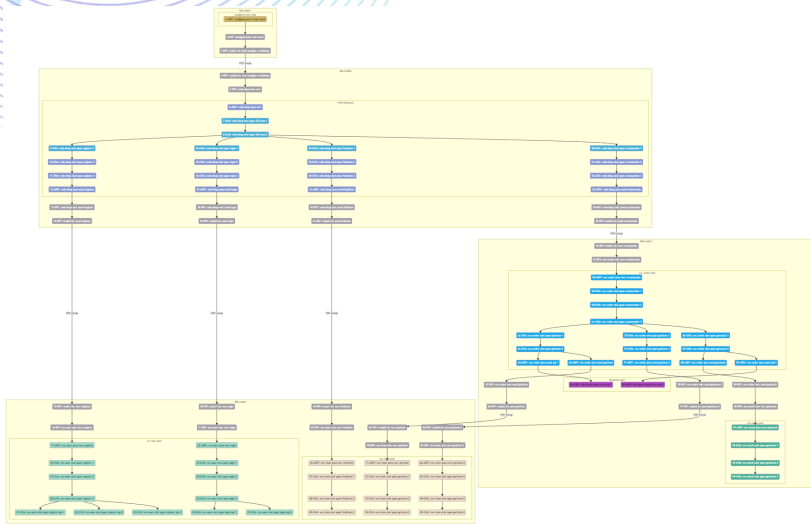
Service List

service_name	duration	duration_ratio
172.17.114.216	30339	67.39
svc-user	5031	11.17
svc-order	4118	9.15
shop-web	3969	8.82
svc-item	1038	2.31
svc-stock	414	0.92
db-demo	57	0.13
loadgenerator	55	0.12

Request Log

```
{
  "_id": "7123452609041598217",
  "related_type": "network/network",
  "start_time_us": 16585580567936,
  "end_time_us": 1658558056781303,
  "duration": 23369,
  "selftime": 0,
  "tap_side": "c",
  "l7_protocol": 20,
  "request_type": "GET",
  "request_resource": "/shop/full-test",
  "response_status": 0,
  "flow_id": "6634231915350067140",
  "request_id": "x_request_id",
  "trace_id": "span_id",
  "parent_span_id": "parent_span_id",
  "req_tcp_seq": 2317163533,
  "resp_tcp_seq": 1939406032,
  "syscall_trace_id_request": 0,
  "syscall_trace_id_response": 0,
  "syscall_cap_seq_0": 0,
  "syscall_cap_seq_1": 0,
  "id": 3,
  "process_id": "vtap_id: 7",
  "service_uid": "service_uid",
  "service_name": "service_name",
  "service_instance_id": "service_instance_id",
  "tap_port": "42:8:83f9d",
  "tap_port_name": "eth0",
  "resource_from_vtap": "resource_from_vtap"
}
```

# 感受 DeepFlow 的无盲点分布式追踪

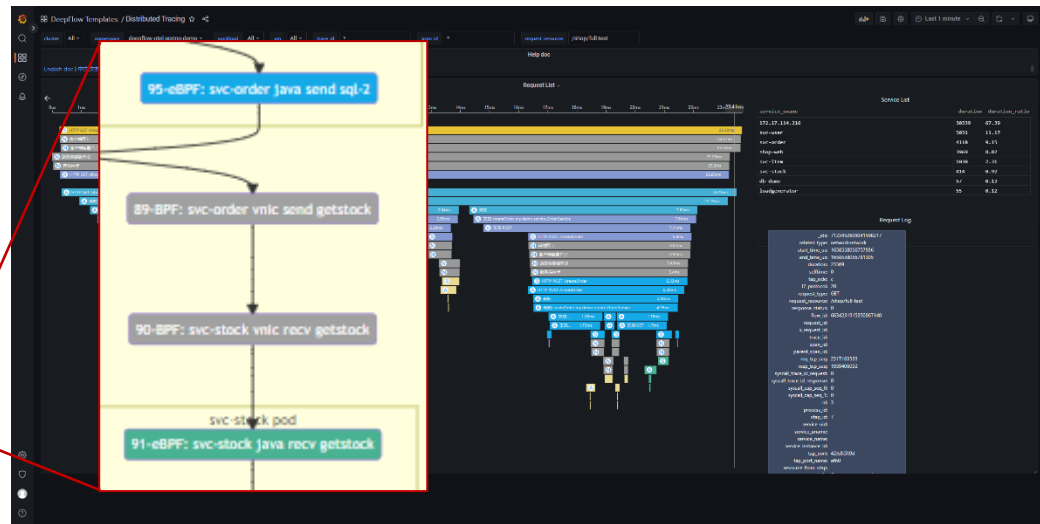
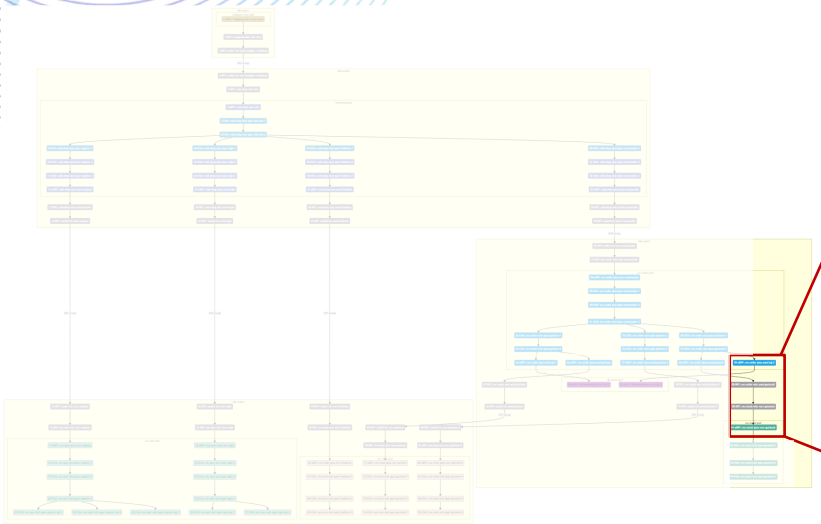


13 个调用，96 个 Span  
46 个 OTel Span (函数调用)  
20 个 eBPF Span (系统调用)  
30 个 BPF Span (网络路径)

1. 全链路：追踪到了 96 个 Span：46 个 OTel Span、20 个 eBPF Span、30 个 BPF Span



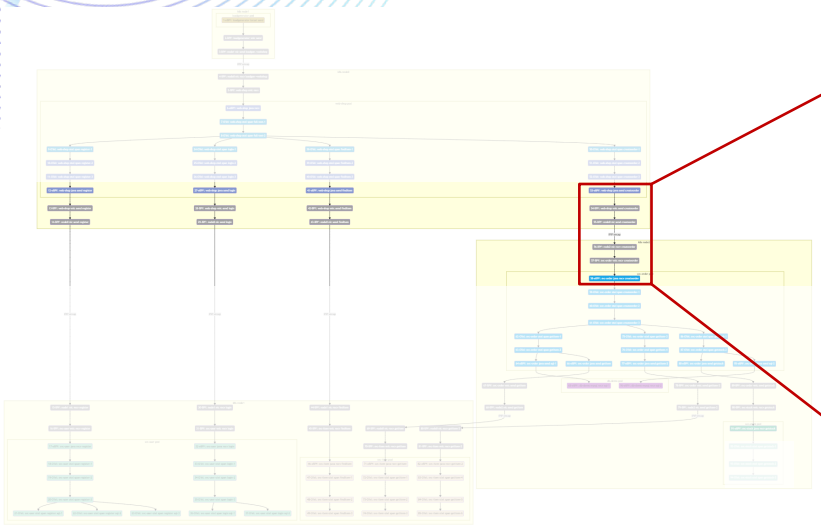
# 感受 DeepFlow 的无盲点分布式追踪



1. 全链路：追踪到了 96 个 Span：46 个 OTEL Span、20 个 eBPF Span、30 个 BPF Span
2. 全栈：支持追踪同 K8s Node 上两个 Pod 之间的网络路径



# 感受 DeepFlow 的无盲点分布式追踪



Request ID	Time	Span	Time	Span	Time	Span	Time	Span	Time	Span	Time	Span
1	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000
2	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000
3	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000
4	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000
5	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000	10:00:00.000

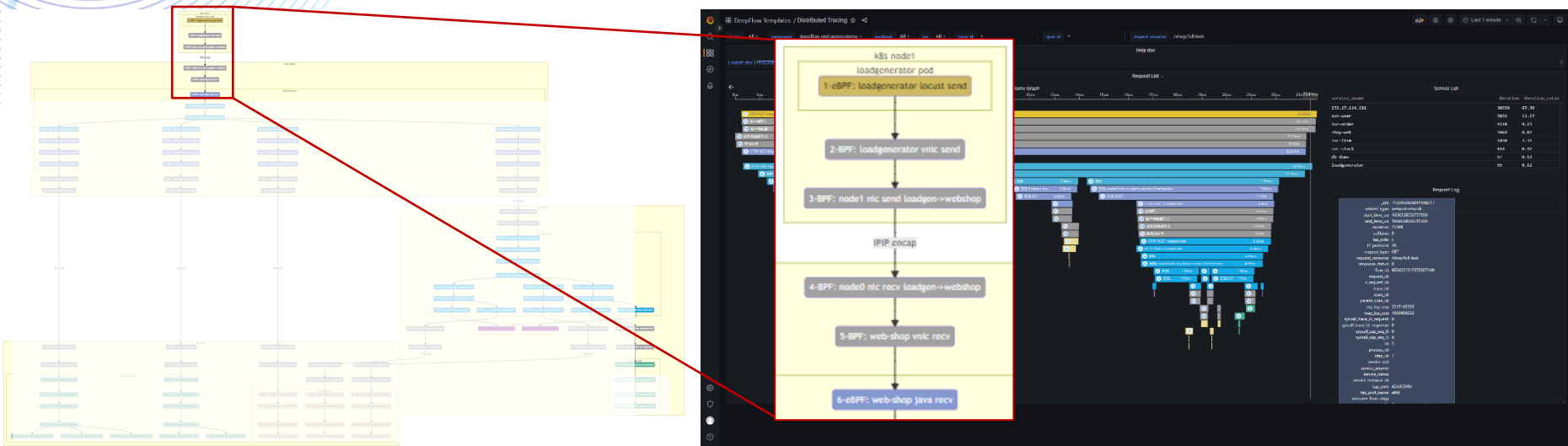
Service Name	Request Count	Response Time (ms)	Success Rate (%)
web-shop	1000	10.17	99.99
svc-order	1000	10.17	99.99
node0	1000	10.17	99.99
node2	1000	10.17	99.99
java	1000	10.17	99.99
vnic	1000	10.17	99.99
nic	1000	10.17	99.99
ipip	1000	10.17	99.99
encaps	1000	10.17	99.99
rcv	1000	10.17	99.99
rcv	1000	10.17	99.99
rcv	1000	10.17	99.99
rcv	1000	10.17	99.99

1. 全链路：追踪到了 96 个 Span：46 个 OTel Span、20 个 eBPF Span、30 个 BPF Span
2. 全栈：支持追踪同 K8s Node 上两个 Pod 之间的网络路径
3. 全栈：支持追踪跨 K8s Node 上两个 Pod 之间的网络路径，即使有隧道

3



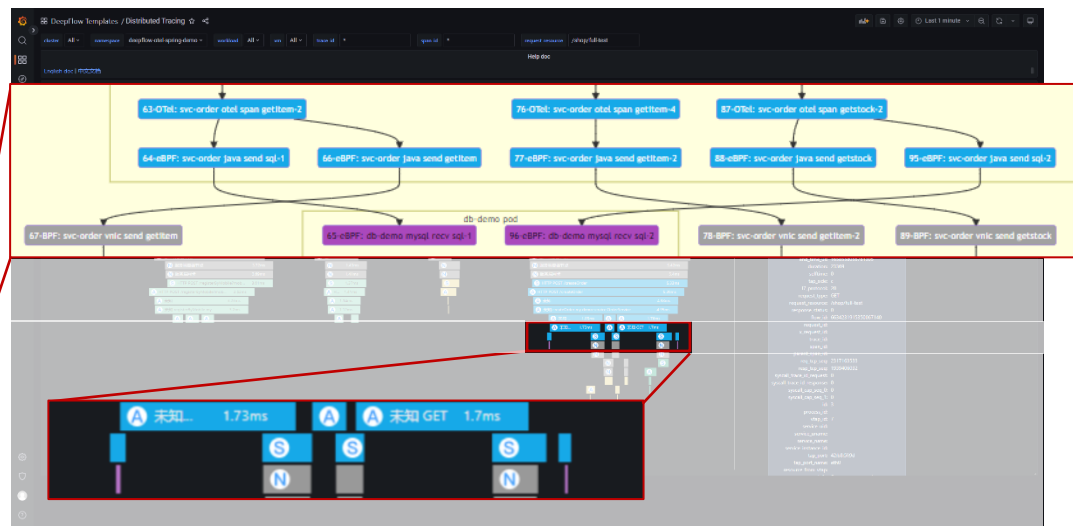
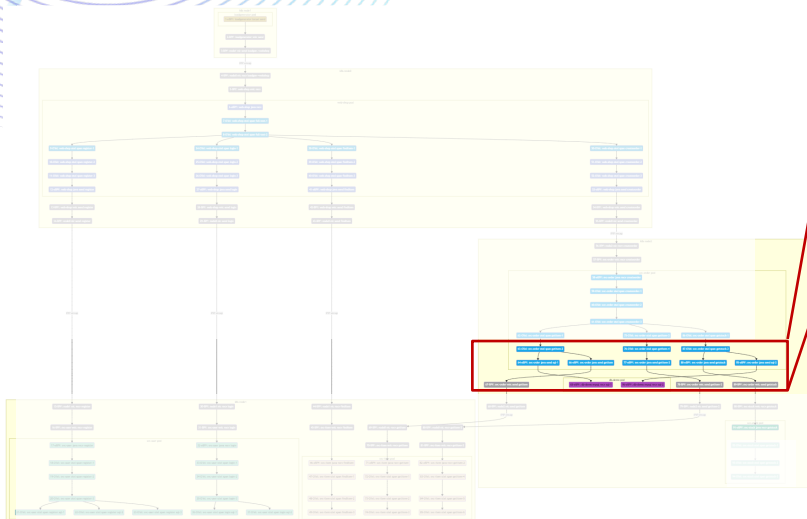
# 感受 DeepFlow 的无盲点分布式追踪



1. 全链路：追踪到了 96 个 Span：46 个 OTEL Span、20 个 eBPF Span、30 个 BPF Span
2. 全栈：支持追踪同 K8s Node 上两个 Pod 之间的网络路径
3. 全栈：支持追踪跨 K8s Node 上两个 Pod 之间的网络路径，即使有隧道
4. 全链路：对 OTEL 无插码的服务（loadgenerator, C），通过 eBPF 自动追踪补齐

4

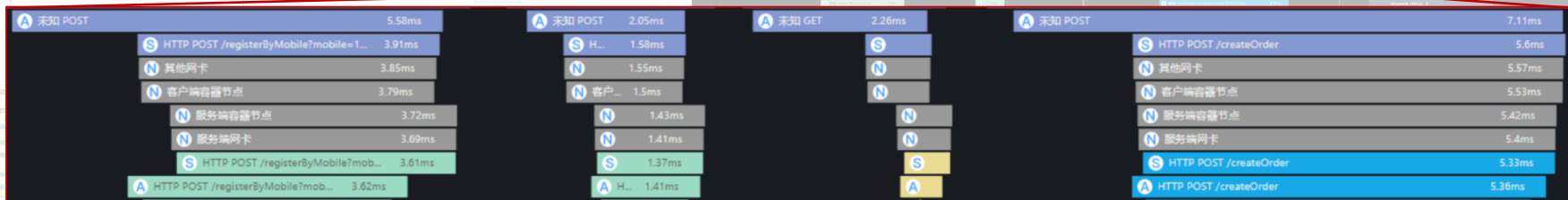
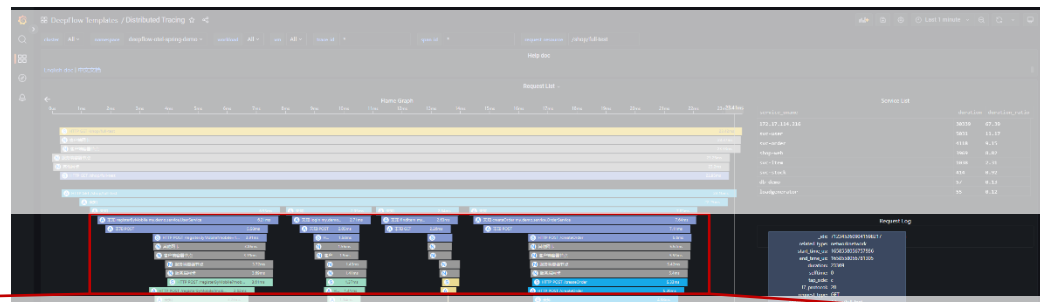
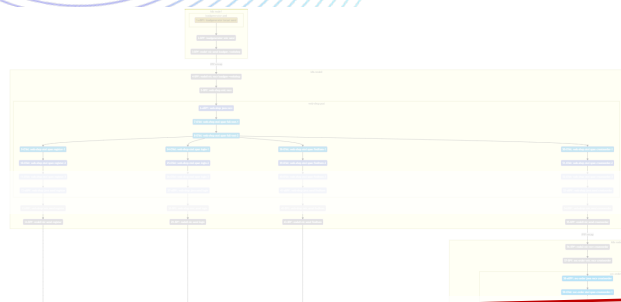
# 感受 DeepFlow 的无盲点分布式追踪



1. 全链路：追踪到了 96 个 Span：46 个 OTel Span、20 个 eBPF Span、30 个 BPF Span
2. 全栈：支持追踪同 K8s Node 上两个 Pod 之间的网络路径
3. 全栈：支持追踪跨 K8s Node 上两个 Pod 之间的网络路径，即使有隧道
4. 全链路：对 OTel 无插码的服务（loadgenerator, C），通过 eBPF 自动追踪补齐
5. 全链路：对 OTel 无法插码的服务（MySQL, autocommit），通过 eBPF 自动追踪补齐



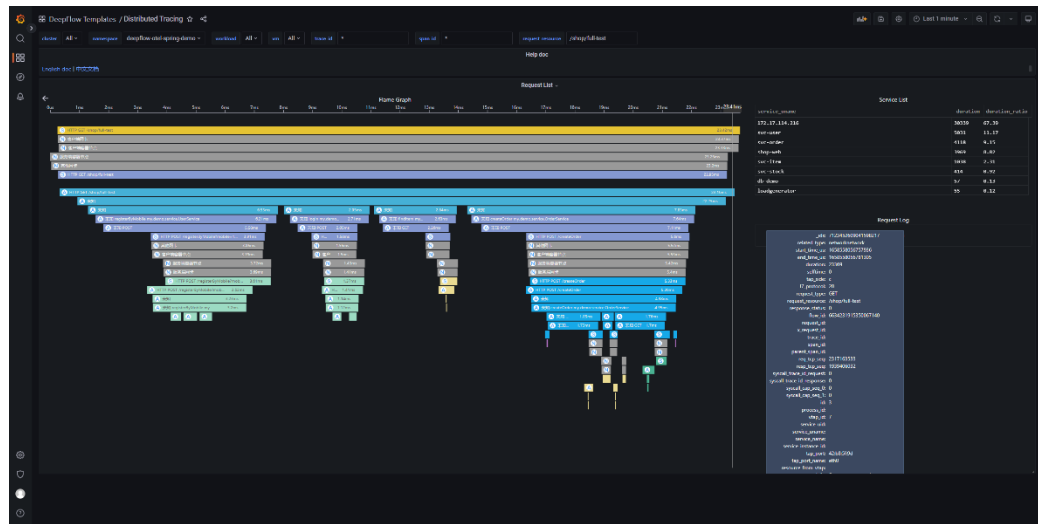
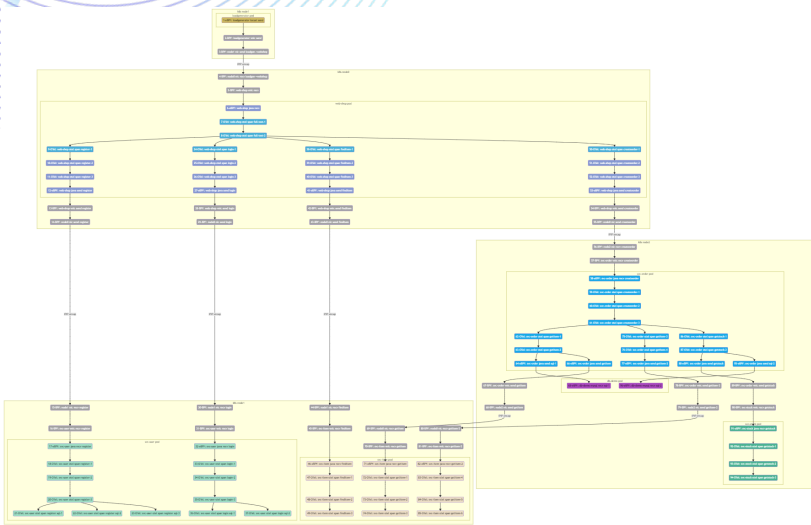
# 感受 DeepFlow 的无盲点分布式追踪



1. 全链路：追踪到了 96 个 Span：46 个 OTel Span、20 个 eBPF Span、30 个 BPF Span
2. 全栈：支持追踪同 K8s Node 上两个 Pod 之间的网络路径
3. 全栈：支持追踪跨 K8s Node 上两个 Pod 之间的网络路径，即使有隧道
4. 全链路：对 Otel 无插码的服务（loadgenerator, C），通过 eBPF 自动追踪补齐
5. 全链路：对 Otel 无法插码的服务（MySQL, autocommit），通过 eBPF 自动追踪补齐
6. 无盲点：eBPF 和 BPF Span 穿插在 Otel Span 之间，让追踪无盲点，满满的协作感 ♥♥♥

6

# 感受 DeepFlow 的无盲点分布式追踪



1. 全链路：追踪到了 96 个 Span：46 个 OTEL Span、20 个 eBPF Span、30 个 BPF Span
2. 全栈：支持追踪同 K8s Node 上两个 Pod 之间的网络路径
3. 全栈：支持追踪跨 K8s Node 上两个 Pod 之间的网络路径，即使有隧道
4. 全链路：对 OTEL 无插码的服务（loadgenerator, C），通过 eBPF 自动追踪补齐
5. 全链路：对 OTEL 无法插码的服务（MySQL, autocommit），通过 eBPF 自动追踪补齐
6. 无盲点：eBPF 和 BPF Span 穿插在 OTEL Span 之间，让追踪无盲点，满满的协作感 ♥♥♥

<https://deepflow.yunshan.net/docs/zh/agent-integration/tracing/opentelemetry/>

# AutoTagging 必然也是稳的

category	keyName	column1_value
知识图谱	pod_cluster_1	k8s-d-rul3j5x3ft
知识图谱	pod_ns_0	deepflow-otel-spring
知识图谱	pod_ns_1	deepflow-otel-spring
知识图谱	pod_node_0	metaflow-demo-1
知识图谱	pod_node_1	metaflow-demo-0
知识图谱	pod_service_0	
知识图谱	pod_service_1	web-shop
知识图谱	pod_group_0	loadgenerator
知识图谱	pod_group_1	web-shop
知识图谱	pod_0	loadgenerator-568b5f6
知识图谱	pod_1	web-shop-5cb6479d7b-d
知识图谱	resource_g10_type_0	10
知识图谱	resource_g10_type_1	10
知识图谱	resource_g10_0	loadgenerator-568b5f6
知识图谱	resource_g10_1	web-shop-5cb6479d7b-d
知识图谱	resource_g11_type_0	101
知识图谱	resource_g11_type_1	101
知识图谱	resource_g11_0	loadgenerator
知识图谱	resource_g11_1	web-shop
知识图谱	resource_g12_type_0	101
知识图谱	resource_g12_type_1	102
知识图谱	resource_g12_0	loadgenerator
知识图谱	resource_g12_1	web-shop

资源申请时  
定义标签

category	keyName	column1_value
标签	label.version_1	latest
标签	label.release_0	
标签	label.release_1	
标签	label.k8s-app_0	
标签	label.k8s-app_1	
标签	label.chart_0	
标签	label.chart_1	
标签	label.heritage_0	
标签	label.heritage_1	
标签	label.istio_0	
标签	label.istio_1	
标签	label.app_0	loadgenerator
标签	label.app_1	web-shop
标签	label.role_0	
标签	label.role_1	
标签	label.name_0	
标签	label.name_1	
标签	label.istio.io/rev_0	
标签	label.istio.io/rev_1	
标签	label.helm.sh/chart_0	
标签	label.helm.sh/chart_1	
标签	label.component_0	
标签	label.component_1	
标签	label.controller-revision-hash_0	

业务上线时  
定义标签

category	keyName	column1_value
原始Attribute	container.id	197b7db26fca4aac843387
原始Attribute	host.arch	amd64
原始Attribute	host.name	web-shop-5cb6479d7b-d
原始Attribute	http.flavor	1.1
原始Attribute	http.host	web-shop:8090
原始Attribute	http.method	GET
原始Attribute	http.route	/shop/full-test
原始Attribute	http.scheme	http
原始Attribute	http.status_code	200
原始Attribute	http.target	/shop/full-test
原始Attribute	http.user_agent	python-requests/2.27.
原始Attribute	k8s.pod.ip	172.17.114.216
原始Attribute	net.peer.ip	172.17.96.103
原始Attribute	net.peer.port	47146
原始Attribute	net.transport	tcp
原始Attribute	os.description	Linux 9.4.105-1.el7.e
原始Attribute	os.type	linux
原始Attribute	process.command_line	/usr/lib/jvm/java-1.8
原始Attribute	process.executable.path	/usr/lib/jvm/java-1.8
原始Attribute	process.pid	7
原始Attribute	process.runtime.description	IcedTea OpenJDK 64-BI
原始Attribute	process.runtime.name	OpenJDK Runtime Envir
原始Attribute	process.runtime.version	1.8.0_212-b04

减少在指标  
中注入标签  
释放研发生产力

# 三步完成! Hello SkyWalking!

## # otel-agent config

```
receivers:  
  # add the following config  
  skywalking:  
    protocols:  
      grpc:  
        endpoint: 0.0.0.0:11800  
      http:  
        endpoint: 0.0.0.0:12800  
service:  
  pipelines:  
    traces:  
      # add receiver `skywalking`  
      receivers: [skywalking]
```

## # otel-agent service yaml

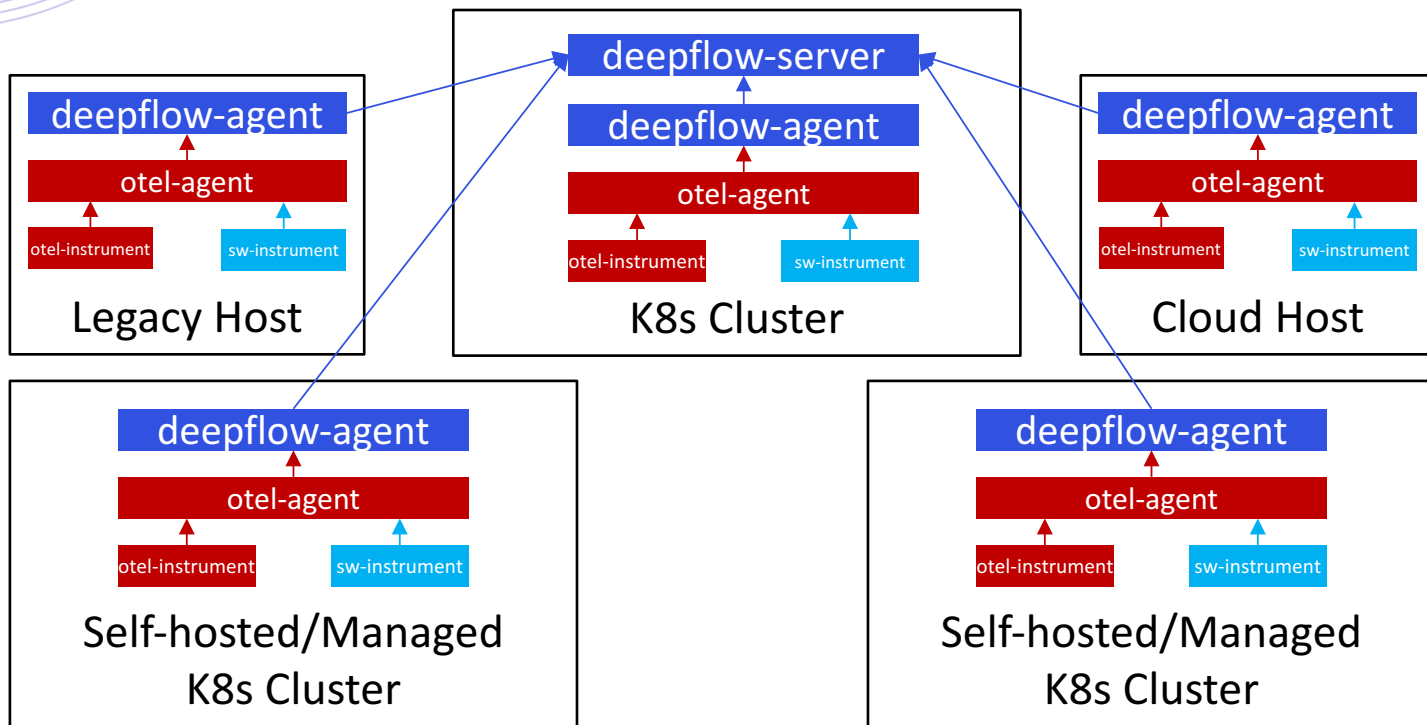
```
spec:  
  ports:  
    - name: sw-http  
      port: 12800  
      protocol: TCP  
      targetPort: 12800  
    - name: sw-grpc  
      port: 11800  
      protocol: TCP  
      targetPort: 11800
```

## # deepflow config

```
vtap_group_id: <your-agent-group-id>  
external_agent_http_proxy_enabled: 1 # required
```

<https://deepflow.yunshan.net/docs/zh/agent-integration/tracing/skywalking/>

# 当然更复杂也没问题，零依赖水平扩展



# 未来迭代的方向

Open  
Telemetry

标准接口

Sky  
Walking

经由Otel集成

Sky  
Walking

直接集成

Sentry

RUM

Grafana  
Tempo

GUI



# 期待大家的互动

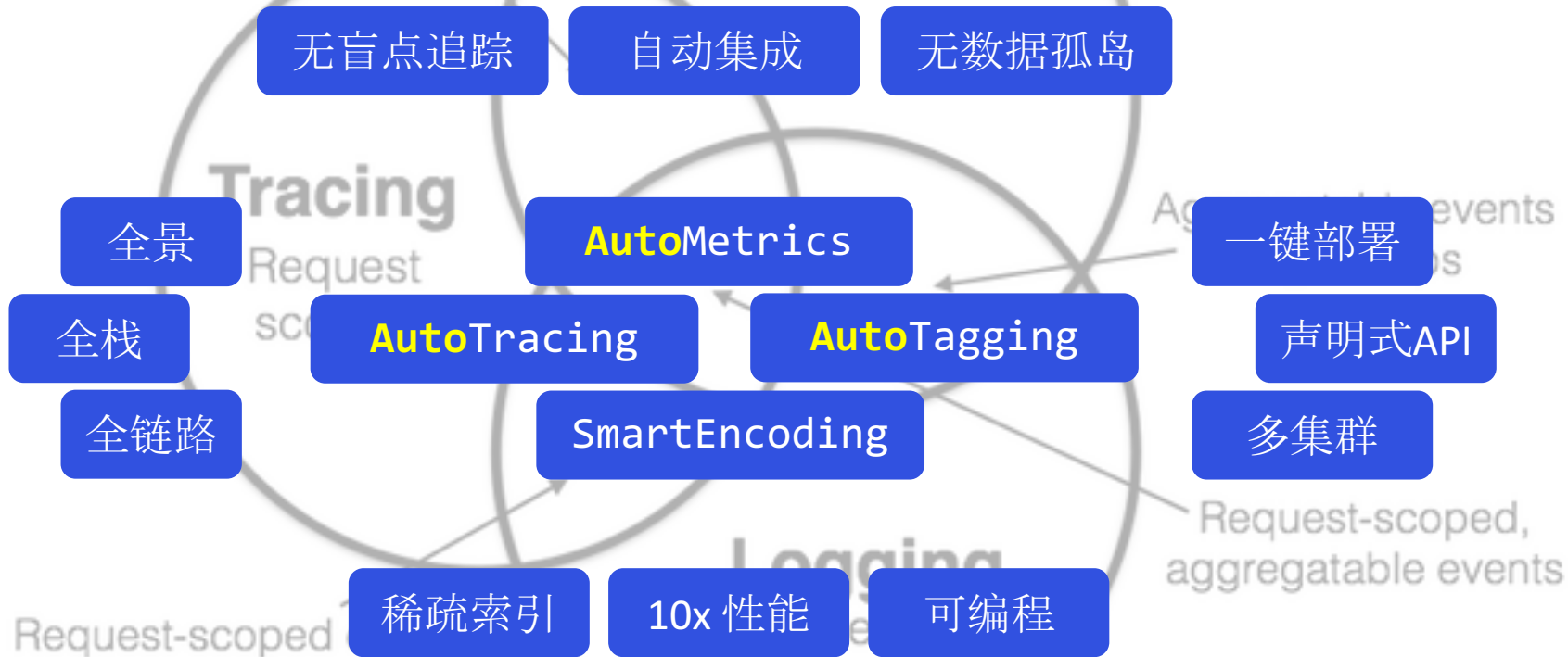
- 目前为止，你感觉 DeepFlow 怎样
- 直播间敲 1，意为“咦，不错哟！”
- 直播间敲 2，意为“啊，真棒呀！！”
- 直播间敲 6，意为“溜，一级棒！！！”



期待 GitHub Star 😊

[github.com/deepflowys/deepflow](https://github.com/deepflowys/deepflow)

# DeepFlow 是一份礼物 希望Cloud-Native Devs、NewOps喜欢



<https://www.youtube.com/watch?v=JUy3GYkPfto>  
"The Future of Ops" by Tyler Treat

# 你好像没有谈到日志

- **BPF**：自动化的全栈性能指标、全景服务关系
- **集成**：自动化的 Prometheus、Telegraf 集成
- **eBPF**：自动化的分布式调用链追踪
- **集成**：自动化的 OpenTelemetry、SkyWalking 集成
  
- **BPF/eBPF**：自动化的网络流日志、应用请求日志、全网包头、全包  
<https://deepflow.yunshan.net/docs/zh/auto-metrics/request-log/>  
<https://deepflow.yunshan.net/docs/zh/auto-metrics/flow-log/>
- **集成**：自动化的 Promtail、Fluentbit、iLogTail 集成 进行中...



DeepFlow®



2022.08.17  
敬请关注

# 【1975年登山路线】

使命：让开发者更自由！

愿景：打造世界级可观测性平台！

从 DeepFlow V6 到 v8，登上最高峰

- AutoMetrics
  - 支持 eBPF+Winpcap 的 multi-OS 能力
  - 基于 eBPF USDT/uprobe 的 C/C++/Java/Golang 等应用 HTTP2/HTTPS 协议解析能力
- AutoTracing
  - 支持更多同步非阻塞调用 (NIO, Non-blocking IO) 场景
  - 支持异步调用 (AIO, Asynchronous IO) 场景
  - 支持协程调度 (hybrid threading) 场景
  - 增强和 OpenTelemetry 的集成能力，通过 eBPF 插入 Otel Tracer API
- AutoTagging & SmartEncoding
  - 同步服务注册中心，自动注入服务和 API 属性信息
  - 非容器环境下自动同步并注入进程标签信息
- Active Probe
  - 支持 Agent 发起主动拨测获取 Metrics
- Continuous Profiler
  - 支持使用 eBPF 采集 On/Off CPU 火焰图，提供零干扰的 Continue Profile 能力
- Agent Programmability
  - 支持 WASM 的可编程应用协议解析能力

- Agent Adaptability
  - 支持 Agent 以 Sidecar 形式运行于 Serverless Pod 中
  - 支持运行于 Android 操作系统中 (智能汽车场景)
- Agent Integration
  - 集成 Sentry RUM 数据源
  - 集成 Promtail/Loki 日志数据源
  - 支持将数据输出至日志文件中，通过日志采集服务 [iLogTail](#) 进行采集
- Server Integration
  - 支持将分布式追踪数据展示在 Grafana Tempo 中
  - 支持将日志数据展示在 Grafana Loki GUI 中
  - 支持 PromQL 作为查询 API
  - 支持作为 Prometheus Exporter 输出 BPF/eBPF 指标数据
  - 支持将数据输出至 OpenTelemetry Collector
  - 支持将数据输出至 Kafka
  - 支持将数据写入至公有云日志存储服务中

# 开始构建可观测性

原力释放 云原生可观测性分享会



Cloud

免费试用



Community

访问 GitHub



Enterprise

咨询专家

我们在招聘，HR 微信：holidayxd