



北京云杉世纪网络科技有限公司

DeepFlow[®]

▶ 让观测更自动

v6.0 用户使用手册

目录

产品介绍..... 11

 产品理念 11

 解决云上业务的网络问题 11

 聚焦对网络流量的采集 12

 混合云网络性能监控的挑战..... 12

 产品功能 13

 功能简介 13

 功能列表..... 13

 软件架构 20

 系统组件..... 20

 系统架构..... 22

 数据流向..... 24

发行注记..... 27

 v6.1.0 27

 应用、网络 27

 系统功能..... 28

 v6.0.1 28

 应用、网络 28

 基础功能..... 29

 系统功能..... 29

 v6.0.0 29

 应用 29

 网络 30

 基础功能..... 30

 系统功能..... 30

v5.7.3	31
全景图	31
基础功能.....	32
系统功能.....	32
v5.7.2	33
全景图	33
基础功能.....	33
系统功能.....	34
v5.7.1	34
全景图	34
基础功能.....	36
系统功能.....	36
v5.7.0	37
全景图	37
包分发	38
基础功能.....	39
系统功能.....	39
v5.6.3	41
全景图	41
包分发	43
基础功能.....	43
系统功能.....	43
v5.6.2	45
全景图	45
包分发	46
基础功能.....	46

v5.6.1	47
包分发	47
全景图	47
流回溯	49
基础功能.....	49
v5.6.0	49
包分发	49
全景图	50
流回溯	50
基础功能.....	50
功能介绍.....	52
总览.....	52
功能使用.....	52
功能规格.....	53
应用/网络.....	53
流量指标.....	53
流量下载.....	165
流量分发.....	167
网络拓扑.....	174
视图.....	176
功能使用.....	176
典型场景.....	181
功能规格.....	187
告警.....	194
功能使用.....	194

功能规格.....	201
报表.....	202
功能使用.....	202
功能规格.....	204
资源.....	204
资源管理.....	204
业务画像.....	212
系统.....	220
功能使用.....	220
典型场景.....	232
功能规格.....	232
二次开发.....	234
应用/网络.....	234
指标数据.....	234
流日志数据.....	239
流量分发策略.....	254
解析分发流量.....	266
网络拓扑.....	270
Grafana 插件.....	270
API 权限获取.....	275
TOKEN 获取.....	275
告警.....	279
告警事件.....	279
系统.....	280
采集器.....	280

故障处理.....	282
控制器告警	282
控制器负载高	282
控制器磁盘空间不足	282
控制器失联.....	283
采集器告警	283
采集器失联.....	283
采集器丢包	284
采集器异常	285
采集器 CPU 超限.....	287
采集器内存超限	287
进程启动.....	287
数据节点告警.....	287
数据节点丢包	287
数据节点负载高	288
数据节点失联.....	288
数据节点磁盘空间不足	289
TSDB 写入失败.....	289
其他告警	290
云平台同步异常	290
慢查询.....	290
页面加载慢.....	290
排查全景图 API 响应慢.....	291
API 错误	294
API 错误.....	294

最佳实践.....	295
推荐配置	295
最小化部署	295
性能规格	295
资源规格	297
集群带宽需求预估	301
通信矩阵	301
控制器与外部通信	301
控制器、数据节点内部通信	302
采集器通信	304
仅发生在一个 K8s 集群内部的通信	305
公有云最佳实践	306
部署形态	306
组网方式	307
资源配置	307
扩容方式	308
腾讯 TCE 最佳实践	308
部署形态	308
组网方式	308
资源配置	309
扩容方式	309
在 TCE 页面的配置	309
阿里专有云最佳实践	309
部署形态	309
组网方式	310

资源配比.....	310
扩容方式.....	310
青云私有云最佳实践	310
部署形态.....	310
组网方式.....	311
资源配置.....	311
扩容方式.....	312
部署扩容.....	312
控制器与数据节点部署.....	312
准备工作.....	312
服务器操作系统安装	313
基础部署.....	328
高可用初始化	341
从区域部署	343
开源模块初始化	344
页面定制配置	345
部署限制.....	354
Kubernetes 部署模式.....	354
虚拟机自动部署模式.....	354
数据节点使用对象存储	356
采集器部署	356
简介	356
采集器部署	356
采集器注册	357
KVM、Workload 和容器类型采集器远程升级	357

采集器批量部署	358
安全传输 [可选]	362
KVM	362
KVM-DPDK	364
Hyper-V	366
ESXi	367
容器	368
专属服务器	375
Workload	389
隧道解封装组件	390
云平台对接	395
软件包获取	395
通用配置	396
FAQ	398
私有云	400
公有云	420
容器	428
其他	430
调试工具	438
扩容	439
简介	439
组件配置文件	439
简介	439
Trisolaris	440
Alarm	442
ZooKeeper	442

Elasticsearch.....	443
开源组件密码配置.....	447
界面配置.....	449
manager.....	451
monitor	452
build.....	452
日常运维操作.....	460
软件升级.....	460
Patch 更新	491
服务器上下线	493
数据备份.....	493
整体卸载.....	494
超时配置.....	495

产品介绍

DeepFlow 是一款云网流量监控产品，面向混合云+容器的全栈虚拟化环境，解决云原生业务的可观测性问题。DeepFlow 深入理解 SDN 网络，依托强大的全网流量采集和全网链路追踪技术，构建云资源的全景网络知识图谱，以全新的资源、微服务、业务视角打开云网黑盒。

DeepFlow 所有软件组件均为云原生架构，采集并沉淀全网流量的 Metrics（性能指标）、Tracing（链路追踪）及 Logging（流量日志）数据，为企业构建跨区域、跨资源池、随云扩展、随云演进的网络数据平台，并向外部 DPI 工具、日志分析系统、调用链追踪系统、大数据分析系统全面开放数据能力。

产品基于应用功能提供应用性能管理（APM, Application Performance Management）解决方案，基于网络功能提供网络监控诊断（NPMD, Network Performance Monitoring and Diagnostics）解决方案，基于流量分发功能提供流量采集分发（NPB, Network Packet Broker）解决方案。

产品理念

解决云上业务的网络问题

DeepFlow 的核心目标是解决云上业务的网络问题。从网络的角度，云网与传统网络的主要区别在于：虚拟化导致的业务 IP 数量 100x 增长、自动化/云原生导致的网络动态性显著增强。为了解决云上业务的网络问题，DeepFlow 通过 API 实现了与主流公有云、私有云、容器管理平台、网络设备之间的信息同步，使得所有流经 DeepFlow 采集器的流量自动关联资源的计算、网络、存储、业务、安全属性。

DeepFlow 监控的资源包括： - **端节点**：Network Endpoint，如云服务器（虚拟机、裸金属服务器）、宿主机、容器 POD、容器节点。 - **网元**：Network Component，如宿主机 vSwitch、容器节点 vSwitch、物理交换机、路由器、NAT 网关、负载均衡器、容器 Service、容器 Ingress。

DeepFlow 围绕被监控资源构建了完整的网络知识图谱，包括： - **计算属性**：IP 对应的云服务器、容器 POD 信息，及所在的宿主机、容器节点资源。 - **网络属性**：资源所在的网络拓扑位置如区域（Region）、可用区（AZ, Availability Zone）、VPC（Virtual Private Cloud）、子网（Subnet），以及所连接网元上相关的网络策略如 VPC 路由、对等连接策略、云企业网策略、NAT 策略、负载均衡策略等。 - **存储属性**：资源提供的 RDS、Redis 等存储服务。 - **业务属性**：资源所在的业务拓扑、所处的业务资源组，及资源组在业务拓扑中的服务依赖关系。一个资源组中的资源通常为一个集群，实现同一项功能。典型的资源组例如负载均衡监听器的后端云服务器集群。 - **安全属性**：在云网络中主要体现为资源直接连接的虚拟防火墙所配置的安全策略，即资源所属的安全组及其安全策略。

DeepFlow 实现了上述资源属性的自动同步，构建了云原生业务的网络知识图谱，并与网络流量的性能监控指标进行自动关联，展现混合云网络的全景图。

聚焦对网络流量的采集

业务监控可通过多种方式实现，例如应用代码插件、应用日志分析、网络日志分析等。作为一个网络监控系统，DeepFlow 聚焦于对网络流量的处理。

网络是连接云原生业务的媒介，对网络流量的采集分析相比其他方式的优势在于：

- **监控网络流量性能指标：**应用代码插件、日志等均不能体现网络层的故障，往往只能给出一个最终的时延或成功与否的结果，网络层面的 TCP 建连时延、重传、零窗等底层原因通过应用代码插件或日志均无法体现。
- **提供更精确的性能数据：**应用日志中记录的性能数据在某些场景下可能存在误差，例如 - Nginx 在响应静态文件请求时，对于不足 1M 的小文件可一次性放入内核缓冲区，这种情况下 Nginx 记录的响应时间远小于网络层面文件实际发送完成的时间，业务环境中二者的差异可高达一个数量级。
- 大多数网络设备支持 sFlow、NetFlow 方式输出采样数据，但这类日志数据一方面采样丧失了精度，另一方面通常仅支持基础的字节数、包数输出，无法输出性能监控数据。
- 在面向互联网提供服务的场景下，网络质量变差时低速客户端会最先失去连接，进而使得服务端的带宽更多的为高速客户端服务。若仅从日志层面分析，应用日志体现出来的现象可能是 QPS（Query Per Second）升高、响应时延降低，表现出“相反”的结论。
- **得到更精准的位置定位：**两个服务之间应用质量的降低，可能由沿途的容器节点、宿主机、网关等多个虚拟网元导致，应用性能监控仅能横向追踪服务调用链路，无法纵向追踪网络虚拟化中的具体网元。

混合云网络性能监控的挑战

混合云环境下流量采集监控的挑战主要来自于业务 IP 数量显著增长和自动化/云原生导致的网络动态性显著增强。虚拟化技术可在一个宿主机上运行 10x 云服务器，容器技术可在一个云服务器上运行 10x POD，最终可导致 IP 数量上升两个数量级。另一方面，云资源的自服务特征、负载均衡以及云原生应用的广泛接纳，使得 IP 地址和资源属性之间的关联关系变动频繁，另外普遍存在的分布式 NAT 也使得人工业务画像不再适用。

这样的变化，从技术层面为网络流量采集监控带来的挑战主要为两个方面：

- **全网流量采集的挑战：**传统网络中通过分光、镜像的方式获取流量，方法成熟。但在虚拟网络环境下，网络的边沿（虚拟交换机）前移到了宿主机及虚拟化的容器节点内，东西向流量的显著增长使得由物理交换机分光镜像无法覆盖全部东西向流量，同时也面临额外镜像端口占用、镜像流量带宽占用的显著开销。因此，混合云环境下的流量采集需要解决全网覆盖和带宽消耗的问题：需要覆盖各种虚拟化、容器环境，并解决 VPC 之间 IP 地址重叠的问题；需要利用遥测技术输出指标数据而非原始流量，仅当有需要时针对性的、触发式的发送特定流量供远端分析工具进一步分析。
- **全栈全链路性能监控的挑战：**云网络的动态性使得业务资源难以沿用传统的 IP 识别方法，云原生业务的分布式特性使得业务的物理及逻辑边界均已消失，如何构建云资源的完整网络知识图谱，使得网络性能监控数据可以从业务视角呈现是

一项巨大的挑战。难点之一在于如何关联混合云下的流量数据，从业务视角展现全栈（服务之间从客户端、容器节点、宿主机、网关到服务端的全路径）、全链路（业务系统的服务间依赖、资源组间依赖、业务间依赖、跨资源池调用、跨可用区调用、跨区域调用）的网络性能数据；难点之二在于云环境固有的规模性和弹性要求监控系统的采集侧、存储侧、控制侧都需要随云扩展、随云演进，以适应十万量级的云资源规模及容器、Serverless 等新型资源池形态；难点之三在于云资源的物理位置动态性、IP 地址动态性和普遍存在的 NAT 地址翻译，使得不再能够通过物理位置或 IP 地址段切割、标记一个业务，需要构建完整的虚拟网络知识图谱；难点之四在于 IP 数量的增多以及业务的分布式趋势，要求故障诊断不能再依靠简单的总量、峰值、均值指标数据，需要提供更丰富的统计计算能力，如百分位数、概率分布、信息熵、标准差等。

产品功能

功能简介

- 应用：微服务应用性能监控，从流量中提取服务的应用性能指标、链路追踪、访问日志数据，并与微服务知识图谱关联，提供应用性能管理解决方案
- 网络：云网络性能监控，从流量中提取服务的网络层性能指标、链路追踪、访问日志数据，并与网络知识图谱关联，提供云网全景视图下的监控诊断解决方案
 - 流量下载：PCAP 数据提供完整的故障回溯取证能力。
 - 流量分发：全网原始流量采集、过滤、截断、去重、标记、分发，提供云网全流量的采集分发方案。
- 基础功能：基础功能对上述解决方案功能进行支撑。
 - 视图：用户自定义统计图表，生成监控大屏、设置告警策略和报表策略的门户。
 - 告警：展示所有告警策略和告警事件。
 - 报表：展示所有报表策略并提供报表下载。
 - 资源：自动或手动同步云网中的资源信息、梳理资源的业务和资源组属性，是全景图网络知识图谱能力的基石。
 - 系统：管理 DeepFlow 平台自身的组件、配置、软件授权、账号、日志等。

功能列表

监控诊断方案 - 应用/网络

功能	功能描述
数据	支持展示物理网络（Underlay 网络）、虚拟网络（Overlay 网络）、容器网络的指标数据
	支持展示物理网络（Underlay 网络）、虚拟网络（Overlay 网络）、容器

	网络的流日志数据、 HTTP 日志数据 、 DNS 日志数据 ，支持记录隧道 IP、负载均衡前客户端真实 IP 等日志信息
	支持过滤下载物理网络（Underlay 网络）、虚拟网络（Overlay 网络）、容器网络的 PCAP 数据
	支持配置 PCAP 存储的过滤策略，可通过 VPC 、子网、云服务器（虚拟机、裸金属服务器）、 IP 、 CIDR 对客户端及服务端进行过滤，可设置网络协议和端口号过滤规则，可设置传输层 Payload 截断规则，支持对 PCAP 文件的流量匹配和存储情况进行监控
知识图谱	支持从 资源池 的视角展示流量访问关系及指标数据，包括： 区域 、可用区、私有云资源池、 容器集群 、 租户
	支持从 计算资源 的视角展示流量访问关系及指标数据，包括： 宿主机 、云服务器
	支持从 网络资源 的视角展示流量访问关系及指标数据，包括： VPC 、子网、路由器、 DHCP 网关 、 IP 地址 、 CIDR
	支持从 网络服务 的视角展示流量访问关系及指标数据，包括： 安全组 、 NAT 网关 、负载均衡器
	支持从 存储资源 的视角展示流量访问关系及指标数据，包括： RDS 、 Redis
	支持从 容器资源 的视角展示流量访问关系及指标数据，包括： 容器集群 、命名空间、容器节点、 Ingress 、容器服务、工作负载（ Deployment 、 StatefulSet 、 ReplicationController ）、 ReplicaSet 、容器 POD
	支持从 业务画像 的视角展示流量访问关系及指标数据，包括： 自定义业务 、自定义资源组、资源组服务
流量属性	支持展示网络流（传输层连接）的 IP 类型（ IPv4 、 IPv6 ）、网络协议、服务端口特征，以及资源之间特定流特征的流量访问关系和流量监控指标数据
	支持展示数据包的播送类型（ BUM 类型 ）、 TCP 标志位 、 TTL 、包长区间特征，以及资源之间特定包特征的流量访问关系和流量监控指标数据
	支持以广域网 IP 所在省份的视角展示流量监控指标数据
指标量	支持展示 网络层吞吐 指标量，包括：字节数、包数、 网络层载荷字节数 、 传输层载荷字节数 、平均包长
	支持展示 传输层吞吐 指标量，包括：新建连接数、关闭连接数
	支持展示 应用层吞吐 指标量，包括： HTTP 请求和应答数 、 DNS 请求和应答数 ，支持解析 HTTP V2 协议
	支持展示 传输层负载 指标量，包括：活跃连接数
	支持展示 传输层时延 指标量，包括： TCP 建连时延 、 TCP 建连客户端时延 、 TCP 建连服务端时延 、 TCP 系统时延 、 TCP/UDP 应用时延
	支持展示 应用层时延 指标量，包括： HTTP 时延 、 DNS 时延 ，支持解析

	HTTP V2 协议
	支持展示 传输层性能 指标量，包括：重传、零窗、重传比例、零窗比例
	支持展示 传输层异常 指标量，包括：TCP 客户端建连失败及其细分原因、TCP 服务端建连失败及其细分原因、TCP 客户端传输异常、TCP 服务端传输异常、TCP 客户端半关、TCP 服务端半关、TCP 超时，支持展示异常连接数所占的比例
	支持展示 应用层异常 指标量，包括： HTTP 4XX 异常、HTTP 5XX 异常、HTTP 服务端超时、DNS 客户端异常、DNS 服务端异常、DNS 服务端超时 ，支持展示异常请求所占的比例
	支持展示 流量资源属性 的指标量，包括：区域个数、可用区个数、宿主机个数、云服务器个数、容器集群个数、容器节点个数、命名空间个数、容器服务个数、工作负载个数、POD 个数、VPC 个数、子网个数、IP 地址个数、CIDR 个数
	支持展示 流量路径属性 的指标量，包括：所有资源属性在客户端侧的个数、所有资源属性在服务端侧的个数
	支持展示 流量自身属性 的指标量，包括： IP 类型（IPv4、IPv6）种数、流经采集点个数、流经采集器个数、网络协议种数、服务端口种数
	支持展示上述吞吐、性能、异常指标量的 峰值、均值、谷值、总量、百分位数、绝对跨度、相对跨度、标准差
	支持展示上述负载、时延指标量的 峰值、均值、谷值、百分位数、绝对跨度、相对跨度、标准差
	支持展示上述指标量相对 指定阈值的百分比
	支持存储上述时延指标量在 1 秒内的最大值，以发现时延的偶发抖动
	支持以网络流（传输层连接）的粒度，展示特定客户端、服务端、协议、客户端端口、服务端端口的上述指标量
全栈链路追踪	支持全栈混合云环境下的虚拟网络全链路追踪，展示在客户端、客户端容器节点、客户端宿主机、客户端网关宿主机、客户端网关、服务端网关、服务端网关宿主机、服务端宿主机、服务端容器节点、服务端等 11 个关键位置的指标数据，以定位虚拟网络端到端路径上的时延、丢包
	支持全栈混合云环境下的物理网络全链路追踪，展示端到端路径在物理链路、物理网元的指标量，以定位物理网络端到端路径上的时延、丢包
	支持在流量日志中关联展示 APM 监控工具的 Trace ID 信息，支持 Datadog、SkyWalking、Zipkin、Jaeger 等 APM，支持自定义能力，实现与 APM 工具的联动
	支持在流量拓扑图中关联展示访问路径及其虚拟链路、物理链路
	支持在流量拓扑图中展示资源之间的 DNAT 关系，包括：容器服务及后端 POD、NAT 网关 DNAT 策略后端云服务器或 IP、负载均衡器及后端云服务

	器或 IP
可 视 化	支持以拓扑图的方式展现流量访问关系及指标数据，支持对拓扑中节点的位置进行记忆保存，支持标记拓扑中资源的关联关系
	支持以排名的方式展现流量访问关系及指标数据，包括柱状图、饼状图、 Top 折线图、堆叠 Top 折线图、表格等形式
	支持以分布的方式展现流量访问关系及指标数据，包括概率分布图、地理位置分布图
	支持以历史趋势的方式展现流量访问关系及指标数据，包括折线图、堆叠折线图、趋势分析柱状图、趋势分析堆叠柱状图
	折线图、堆叠折线图、 Top 折线图、堆叠 Top 折线图支持设置滑动时间窗口大小以实现监控数据的平滑展现
	支持展示云资源的变更事件日志，包括云服务器的创建（同步）、删除、启动、停止、迁移、 IP 变化事件，支持将变更事件与监控指标数据关联展示
	支持展示容器资源的变更事件日志，包括 POD 的创建（同步）、删除、重建、迁移、 IP 变化事件，支持将变更事件与监控指标数据关联展示
	支持展示网络流（传输层连接）的详细日志信息
	上述所有可视化图表，支持添加到自定义视图、下载 CSV 数据、查看 API、全屏显示、弹框展示、样式设置
	上述所有可视化图表，均可设置展示全景图所支持的指标量，并可设置指标量阈值、差量阈值，并支持对折线图设置 基线阈值
	上述所有可视化图表，均可设置展示指标量所查询的数据源时间粒度
搜 索 能 力	支持通过资源知识图谱信息、流属性、包属性、广域网属性对流量进行搜索过滤
	支持对搜索结果按资源知识图谱维度进行分组聚合展现
	支持对搜索结果自动按最精细的计算、网络、存储、容器资源进行分组聚合展现
	支持对搜索条件进行记忆保存，支持将搜索历史分享给其他账号
	搜索条件支持引用模板变量，实现搜索条件快速切换
网 络 拓 扑	支持以 VPC、子网、云服务器、宿主机等方式查询虚拟网络逻辑拓扑
	支持检测指定端到端云服务器、 IP 之间的可达性，包括由于安全组、路由、对等连接、云企业网等造成的可达性问题

采集分发方案-流量分发

功能	功能描述
流量	支持以源 IP、目的 IP、协议、源端口号、目的端口号的五元组方式配置分

分发	发过滤策略
	支持源端、目的端以云服务器、VPC、子网、IP 端、容器 Deployment、容器 StatefulSet、容器服务等方式配置分发过滤策略
	支持源端、目的端以自定义 IP 资源组、自定义云服务器资源组等方式配置分发过滤策略
	分发策略支持跟随云服务器迁移
	分发策略支持跟随容器 POD 重建、Deployment 扩容缩容
	分发策略支持根据子网中云服务器的新建和删除自动动态调整
	分发策略支持设置 Payload 截断长度
	支持对分发策略的流量匹配情况进行监控
	支持对分发流量标记外层隧道标签和内层 VLAN 标签
	支持对分发流量标记采集时间戳
	支持对分发流量标记流量采集的方向
	支持对分发流量实现全局去重，例如两个云服务器之间通信的流量只会发送一份到后端分析工具；支持控制去重机制的打开和关闭
	支持多路分发，流量可分发至多个隧道端点
	分发流量支持 VXLAN、ERSPAN 隧道封装格式
	支持对分发流量进行隧道解封装后转发到分析工具

基础能力支撑

功能	功能描述
视图	支持将全景图中的子视图添加到自定义视图
	支持对子视图的尺寸、位置进行自定义排布
	支持创建模块以容纳多个子视图，支持对模块进行折叠以加速页面加载
	支持设置监控大屏风格的视图
	支持对折线图设置邮件告警策略
	支持对整个视图设置天、周、月报表
	支持导出、导入视图
	支持将视图分享给其他账号
告警	支持对监控平台自身的负载、丢包、CPU/内存/硬盘使用率、进程重启、连通性等异常进行告警
	支持对云平台同步异常事件进行告警
	支持对全景图中的 Top 折线图和折线图设置告警策略
	支持通过跨度、标准差算子对指标量的陡增陡降、毛刺进行告警

	支持通过滑窗对告警事件进行抑制，避免抖动的指标量统计值频繁触发告警
	支持展现和查看告警事件详情
	支持通过邮件发送告警事件
	支持通过 HTTP API 推送告警事件至外部告警平台或工作流系统
	支持通过告警事件动态触发 PCAP 数据的存储
报表	支持生成离线 HTML 报表
	支持通过邮件定时推送报表
资源	支持与主流公有云进行资源同步，包括： AWS、阿里云、腾讯云、华为云、青云、微软云
	支持与主流私有云/专有云进行资源同步，包括： 阿里专有云、腾讯 TCE、青云、金山银河云、华为云、微软 ACS、ZStack、OpenStack、vSphere、NSP
	支持与主流容器管理平台进行资源同步，包括： Kubernetes、腾讯 TKE、阿里 ACK、QingCloud QKE/KubeSphere、OpenShift
	支持关联容器资源与底层承载它的云资源知识图谱信息
	支持设置云平台、容器集群的资源同步时间间隔，支持对资源同步的耗时进行监控
	支持与物理网元进行资源同步，包括： F5
	支持同步云平台的资源池信息，包括： 区域、可用区、租户
	支持同步云平台的计算资源信息，包括： 宿主机、云服务器
	支持同步云平台的网络资源信息，包括： VPC、子网、路由器、IP 地址
	支持同步云平台的网络服务信息，包括： 安全组、NAT 网关、负载均衡器、对等连接、云企业网
	支持同步云平台的存储服务信息，包括： RDS、Redis
	支持同步容器集群的信息，包括： 集群、命名空间、节点、POD、ReplicaSet、工作负载（Deployment、StatefulSet、ReplicationController）、服务、Ingress
	支持用户自定义业务画像，并在业务中创建资源组及其服务
	支持创建 IP、云服务器、容器服务、容器工作负载、负载均衡后端主机类型的资源组
	支持录入物理网元、物理采集点、物理链路
	支持对云平台、资源组、物理网元的图标进行管理
租户	支持 创建租户 ，并授予 VPC、容器命名空间查看权限
	租户支持查看全景图、视图、告警、报表、资源功能

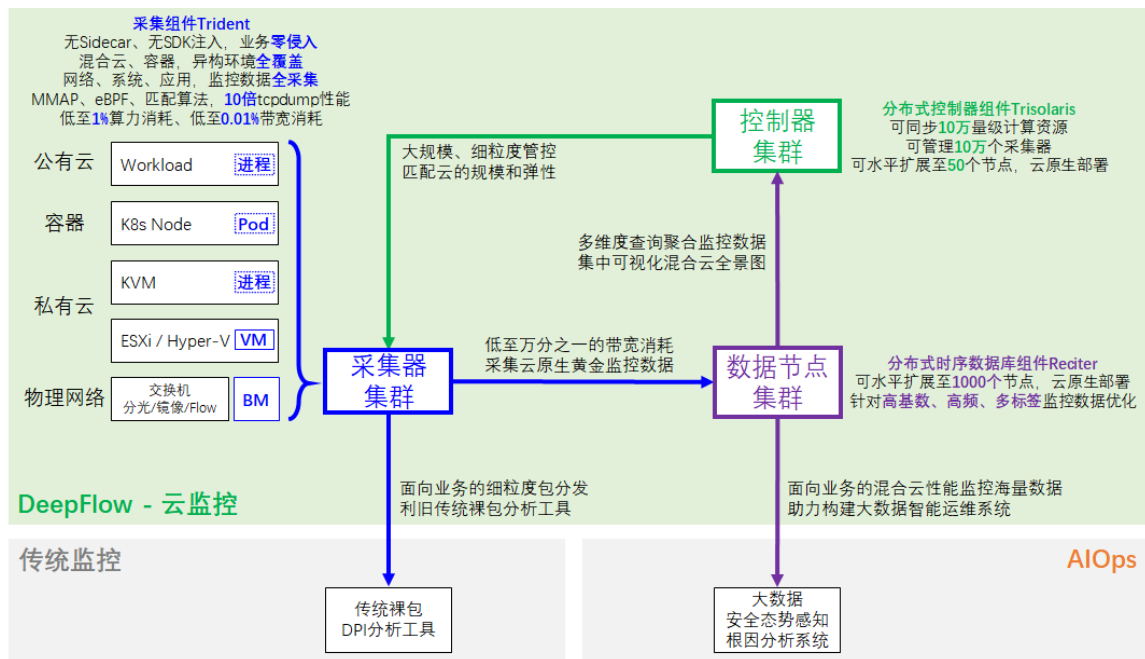
	所有账号之间支持以只读或读写的方式共享全景图搜索历史、视图
	全景图、资源页面支持通过租户过滤资源，全景图知识图谱支持显示租户信息
控制器	支持主备高可用自动切换，支持水平扩展，支持导入导出控制器关键配置
	支持分布式管理、集中呈现多数据中心（区域/Region）、多资源池（可用区/AZ）的控制器集群、数据节点集群、采集器集群
	支持设置每个控制器关联的区域、可用区，控制器会将采集器自动均衡的关联到本区域、可用区的控制器上
	支持设置每个控制器可最多关联的采集器，避免在高可用切换时造成控制器过载，实现对控制器的过载保护
	支持对控制器设置运维状态进行维护
	支持配置 NTP 服务器、LDAP 服务器、RADIUS 服务器、Windows AD（OpenLDAP）服务器
	支持对控制器的 CPU、内存、磁盘、负载进行监控
	支持管理 NAT 网关连接的采集器、数据节点
	支持运行于 X86、ARM 体系架构下
采集器	支持以无依赖的用户态进程方式运行在 KVM 宿主机上，对所在宿主机上所有虚拟机的流量进行采集
	支持以无依赖的用户态进程方式运行在用户 Workload（云服务器）上，对 Workload 的流量进行采集
	支持以 Daemonset 的方式运行在 K8s 节点上，对所在节点上所有 POD、NAT 转换网口、节点网口的流量进行采集
	支持以独立虚拟机的方式对 vSphere、Hyper-V 宿主机上的所有虚拟机流量进行采集
	支持运行于 Open vSwitch Kernel、Open vSwitch DPDK、Linux Bridge 等宿主机、容器节点的网络环境中
	支持运行于 Flannel、Calico、腾讯 TKE GlobalRouter/VPC-CNI 共享网卡、阿里 ACK Terway、青云 QKE HostNIC 等容器网络环境中
	支持运行于 Linux、Windows 操作系统环境中，Linux 环境下采集器不监听任何服务端口、不引入任何依赖包
	支持在控制器上跨区域管理所有采集器，对采集器进行批量远程启用、禁用操作
	支持对所有采集器的启动时间、策略同步时间进行监控
	支持对采集器关联的控制器、数据节点进行全局均衡
	支持对采集器进行分组，并对每个分组设置过载保护策略，包括：CPU 限

	制、内存限制、采集速率限制、分发速率限制、分发熔断阈值、策略同步时间间隔
	支持对采集器的资源消耗进行监控，包括：CPU 用量、内存用量、运行环境系统负载、采集流量、采集包量、分发流量、分发包量、丢包情况
	支持对采集器采集的虚拟网卡流量进行监控
	支持对控制器的 逃逸时间 进行配置，采集器无法连接控制器时将会以最后接收到的策略维持运行，并在运行指定时长仍未与控制器建立联系之后自动禁用
	支持与控制器通过 安全信道（SSL） 进行通信
	支持与控制器、数据节点通过 IPv6 地址进行通信
	支持解析云平台、容器集群的 Underlay 隧道封装流量，包括： VXLAN 、腾讯 TCE GRE 、 Calico IPIP
	支持解析虚拟化+容器全栈环境的双层隧道封装流量，包括： VXLAN-VXLAN 、 VXLAN-IPIP
	支持将隧道封装的 Underlay 流量与 Overlay VPC 流量进行关联，并支持不同 VPC IP 地址冲突 的场景
	支持运行于 X86 、 ARM 体系架构下
数据节点	支持 水平扩展 、 跨数据中心部署
	支持设置每个数据节点关联的区域、可用区，控制器会将采集器自动均衡的关联到本区域、可用区的数据节点上
	支持设置每个数据节点可最多关联的采集器，避免在高可用切换时造成数据节点过载，实现对数据节点的过载保护
	支持对控制器设置运维状态进行维护
	支持对全景图指标数据设置保留时长
	支持自定义生成小时、天粒度的全景图指标量数据
	支持运行于 X86 、 ARM 体系架构下
其他	支持管理监控平台账号
	支持查看和下载操作日志和系统日志

软件架构

系统组件

DeepFlow 由控制器、数据节点、采集器三大组件构成。



系统组件

控制器

控制器是 DeepFlow 的大脑，负责管理所有的采集器和数据节点，控制器集群可扩展至 50 个节点，全局可管理 10 万个采集器。整个集群中的控制器分为三种角色：

- 主控制器：整个 DeepFlow 平台的控制中枢和提供对外交互、服务的接口，也负责与云平台、容器平台、CMDB 等外部系统进行资源信息同步。DeepFlow 平台中只有一台主控制器，主控制器所在的区域（Region）称之为主区域。
- 备控制器：与主控制器的能力完全一致，当主控制器出现宕机或不能提供服务的其他故障时，自动切换为主控制器。在没有备控制器的情况下，DeepFlow 控制器集群没有高可用能力。整个 DeepFlow 中只有一台备控制器，且必须和主控制器在同一个区域中，并共享同一个用于提供外部服务的虚 IP 地址。当主区域内仅有 2 台控制器时 DeepFlow 可提供主备控制器的手动切换能力，当主区域内控制器数量大于等于 3 台时可提供自动化的主备控制器高可用能力。
- 从控制器：负责控制所在区域或可用区（AZ，Available Zone）中的采集器及数据节点，与主控制器保持信息同步，并将主控制器的策略和云平台资源信息同步至关联范围内所有的采集器和数据节点。除主区域外，每个区域中至少部署一台从控制器，同一个区域的多台控制器之间可以实现负载均衡和自动高可用切换。

区域主节点：区域内执行部署命令和保存部署相关配置的控制器，每个区域有且仅有一个。主区域主节点是主控制器，从区域主节点为部署时选定的一个从控制器。

采集器

采集器是 DeepFlow 的自研高性能流量采集器软件 Trident，其采集性能为 tcpdump 的 10 倍，计算资源消耗可低至运行环境总资源量的 1%、输出指标数据

时的带宽消耗可低至业务流量的 0.01%。采集器支持运行于物理网络、虚拟化、容器等多种环境中：- **KVM**：运行于 KVM 宿主机上，负责采集宿主机上所有虚拟机的流量。- **KVM-DPDK**：运行于 OvS DPDK 环境的 KVM 宿主机上，页面展现的采集器类型与 KVM 相同。- **ESXi**：运行于 vSphere ESXi 上的一台专用采集器虚拟机中，负责采集所在 ESXi 主机上所有其他虚拟机的流量。- **Hyper-V**：运行于 Hyper-V 宿主机上，负责采集其上所有虚拟机进出宿主机的流量。- **容器-V**：运行于虚拟化的容器节点上，负责采集所在容器节点上所有 POD 的流量。- **容器-P**：运行于物理容器节点上，负责采集所在容器节点上所有 POD 的流量。- **专属服务器**：运行于一台专属的服务器中，负责采集物理交换机通过镜像、sFlow、NetFlow/NetStream 等方式发送的流量。- **隧道解封装**：运行于虚拟机或裸金属服务器中，实现对采集器分发流量的 VXLAN 解封装，使得流量可直接分发给第三方分析工具。- **Workload-V**：运行于虚拟机中，负责采集所运行的 Workload 虚拟机的流量。- **Workload-P**：运行于传统物理服务器、云环境裸金属服务器中，负责采集所运行的 Workload 裸金属服务器的流量。

另外，专属服务器类型的采集器会将 PCAP 数据存储于所在服务器的文件系统中，为流量下载功能提供数据。

数据节点

数据节点是 DeepFlow 提供全景图功能的必要组件，支持水平扩展至 1000 个节点。主要职责包括：- **时序数据存储**：接收采集器发送的指标数据和流量日志数据并存储于分布式数据仓库中，为全景图流量搜索功能提供数据。- **PCAP 存储**：接收采集器发送的压缩包并以 PCAP 形式存储于文件系统中，为全景图流量下载功能提供数据。

系统架构

整体架构

DeepFlow 是面向多区域混合云的统一平台，支持以分布式的架构部署于多区域、多可用区的环境中并实现统一管理。每个区域中的采集器由该区域的控制器进行本地管理，且将采集数据发送到该区域的数据节点中。若区域中不同可用区之间有严格的防火墙控制，划分粒度可进一步精细到可用区。同一个区域内部所有数据节点之间可进行数据的聚合计算，并由主控制器汇总所有区域的结果数据。



整体架构

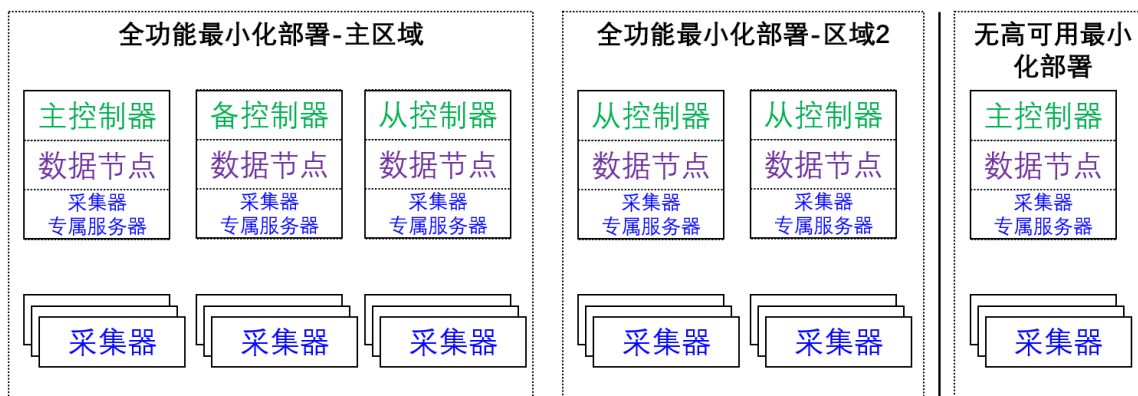
高可用

在高可用场景下，主区域（主控制器所在区域）中一般部署主控制器和多个从控制器。为了实现自动化的主备控制器高可用切换能力，主区域中控制器数量至少需要 3 台，组成稳定的 ZooKeeper 集群用于主备控制器选举。当某个控制器发生故障时，数据节点自行切换请求区域/可用区内的其他控制器。考虑到采集器 10 万级别的庞大量级，主控制器会均衡分配每个控制器和数据节点负责响应的采集器列表。主控制器对所有控制器和数据节点的健康状况进行实时探测，当控制器或数据节点发生故障时，主控制器协调与之关联的采集器进行自动切换。

数据节点运行 ES 集群和时序数据库集群，各区域的 ES 集群、时序数据库集群相互独立，以避免跨区域交换数据。ES 集群高可用采用 Elasticsearch 本身的机制。时序数据库集群依托开源 ClickHouse 构建，高可用及水平扩展采用 ClickHouse 本身的机制。当某个区域内存在大于一台数据节点时即可实现该区域的时序数据高可用存储，每份数据会存储于区域内的两个数据节点上。当任何一个数据节点故障时，其上的数据写入请求将会被均衡迁移至区域内其他数据节点，故障期间的数据读取不会受到影响，数据节点恢复后故障期间的数据也将自动恢复为双份存储。同样，如果区域内不同可用区无法通信，集群的粒度也可细化至可用区内。

最小化部署

DeepFlow 系统组件均为可云化部署的软件，精简情况下不同的系统组件可部署于一台虚拟或物理服务器上。



最小化部署

主区域中使用三台服务器可部署一套最小化的全功能 DeepFlow。每台服务器承担控制器、数据节点、专属服务器类型的采集器角色。其中专属服务器类型的采集器用于接收物理网络镜像流量和 sFlow/NetFlow/NetStream 数据。其他采集器部署于 KVM/ESXi/Hyper-V/容器/Workload 环境中，按实际数量需求部署即可。主区域之外，每个区域的最小化全功能部署需要两台服务器，承担从控制器、数据节点、专属服务器类型的采集器角色。

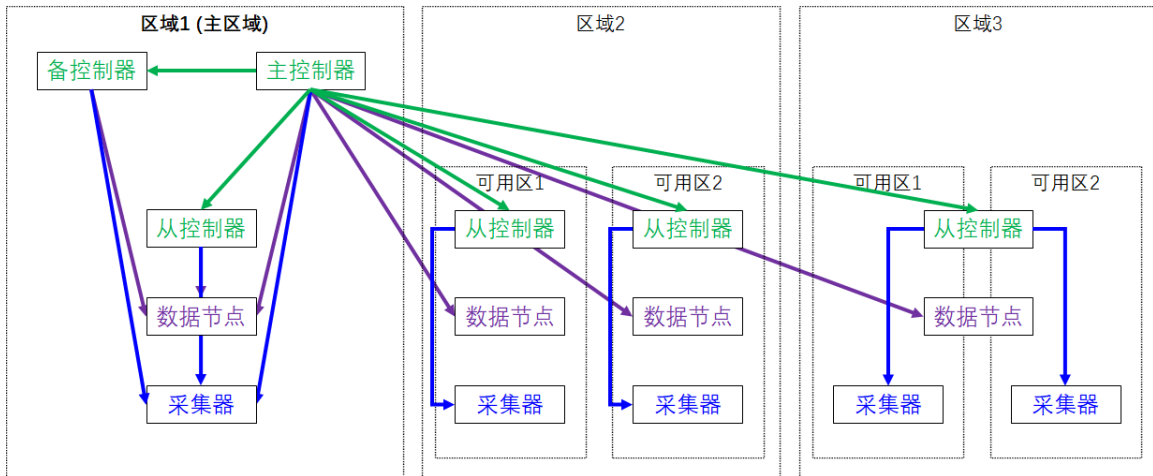
若无需高可用能力，DeepFlow 的极简模式支持以 ALL-IN-ONE 的方式将主控制器、数据节点、专属服务器类型的采集器部署于一个服务器上。

数据流向

DeepFlow 是一个云化的监控数据平台，其中的数据流向遵循本地优先原则，最大限度的降低跨可用区、跨区域交互，同时达到统一管理的目标。

集群控制

集群控制（策略下发）以主控制器为中心，以从控制器为扩展，实现对所有数据节点、采集器的分区域、分可用区同步机制。



集群控制

每个区域中的采集器可请求本区域的控制器实现策略同步，同时用户也可限定控制器只响应本可用区的采集器请求，以进一步降低同一个区域内跨可用区的流量。

除此之外，组件之间的时间同步、部署升级等均限定在一个区域内部。

流量分发

流量分发由控制器下发策略，由采集器自主完成。

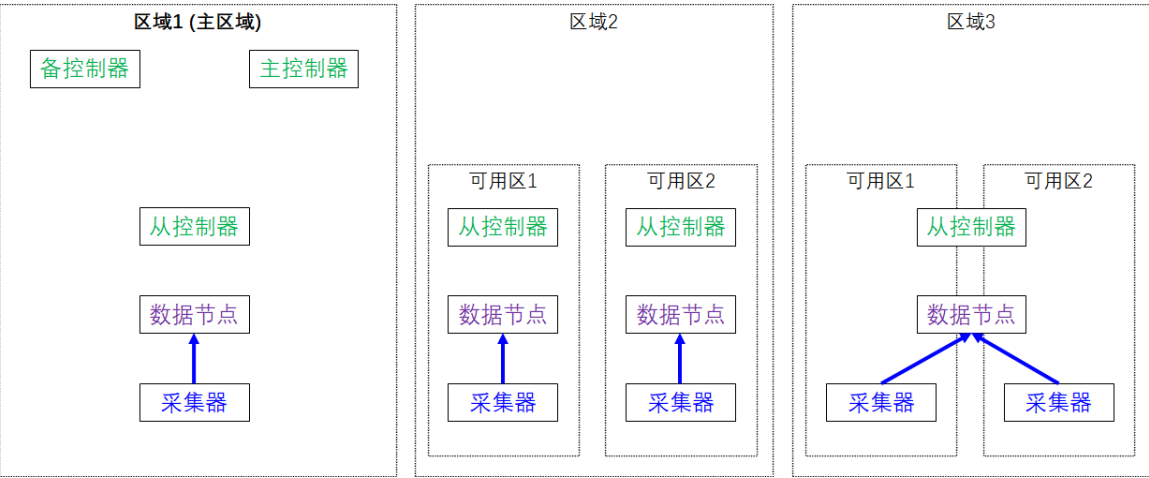


流量分发

采集器对流量进行过滤、去重、截断、标记之后直接封装在隧道中发送给隧道对端的解封装工具。DeepFlow 提供 VXLAN 隧道的解封装采集器，同时也支持发送到 TAP 交换机、汇聚分流等硬件设备上上进行隧道解封装。除了在原始流量外层封装隧道头部以外，采集器还会在原始流量中插入一个 VLAN 头部，记录隧道标签和方向标签，使得隧道解封装设备仅需执行简单的隧道头部剥离即可，无需感知隧道标签和方向标签。

数据采集

采集器对流量进行处理，并将处理后的结果发送至数据节点。

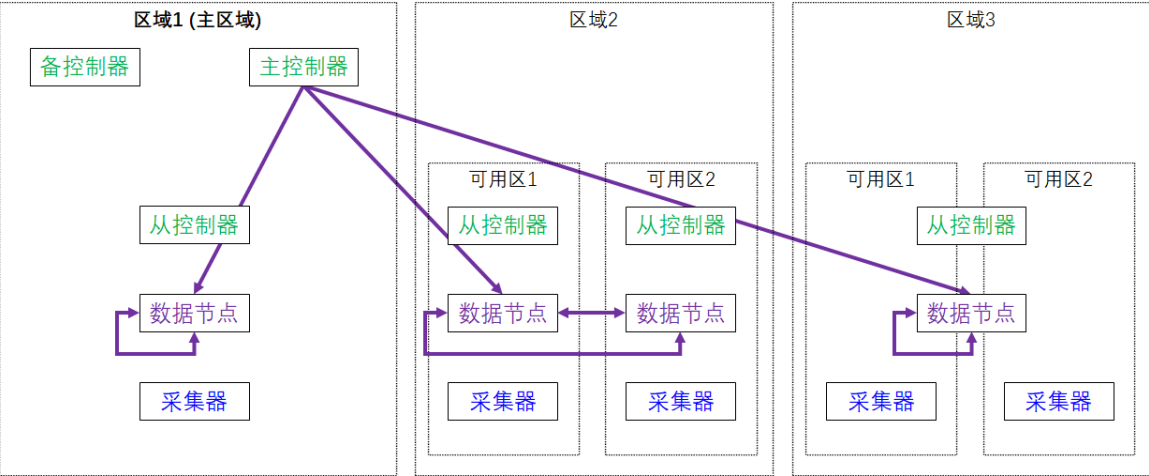


数据采集

采集器可采集指标数据、流量日志数据、PCAP 三类数据，用于全景图流量搜索、流量下载功能。指标数据和流量日志数据就近存储于本区域的数据节点集群中，PCAP 数据就近存储于本区域的数据节点和专属服务器类型的采集器中。

数据检索

数据检索由主控制器发起，通过调用数据节点的 API 完成。下图主要描述指标数据在检索中的流量。



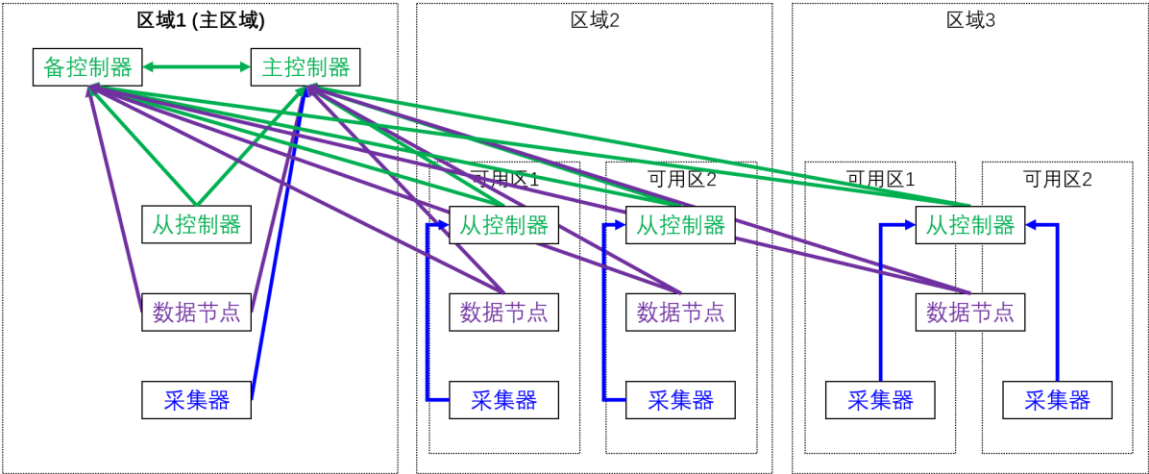
数据检索

区域内的数据节点进行本地聚合（Map）之后与同区域内其他数据节点交换（Shuffle）中间数据，并由数据节点完成最终聚合（Reduce）后返回给主控制器。主控制器收集各个区域的返回数据，排序、截取后形成最终结果进行展现。

当一个区域内的不同可用区之间有严格的防火墙限制时，为了最大限度的降低开墙工作量，可以仅将其中一个可用区（记为 A）中的数据节点用于数据聚合。这种情况下，只需要将 A 可用区与其他可用区之间的通信放行，无需放行所有可用区之间的防火墙，实现开墙工作量 $O(N^2)$ 到 $O(N)$ 的降低。由于此时除了可用区 A 之外的其他可用区不参与聚合，实际上也不会向主控制器返回查询数据，如上图区域 2 的可用区 2。

自身性能监控

DeepFlow 本身即是一个分布式系统，通过采集所有组件的性能数据对自身进行实时性能监控。



性能监控

控制器、数据节点的性能监控数据直接发送至主备控制器，非主区域的采集器性能监控数据经由从控制器转发至主备控制器。

除此之外，所有软件组件（包括采集器）的日志将会就近存储于本区域的 ES 集群中，所有数据节点自身的 APM 链路追踪监控数据将会汇总存储于主区域的 ES 集群中。

发行注记

v6.1.0

应用、网络

- 应用：调用链追踪
 - 支持利用 eBPF 技术自动追踪分布式应用调用链，目前支持 Kernel-Level 线程模型及同步调用场景
 - 支持自动关联 eBPF、BPF 采集的应用调用数据，实现从应用到基础设施的全链路追踪

- 支持自动关联 HTTP/Dubbo 协议中的 TraceID、SpanID、RequestID 字段，增强调用链追踪
- 网络：NAT 追踪
 - 支持 SNAT、DNAT、FULLNAT 前后流量追踪
 - 支持自动追踪一次或多次 NAT 前后的网络全路径
 - 支持腾讯专有云（TCE）中 NFV 网关区流量的全路径追踪
- 易用性改进
 - Grafana：增加交互式的搜索条件编辑器
 - 快速搜索：应用调用详情、网络流日志详情页面提供易于使用的快速搜索框
 - 菜单合并：合并所有应用协议二级菜单为调用，合并右滑框中的应用调用详情表格

系统功能

告警

- 增加采集器运行环境中系统内存空闲比例告警
- 增加采集器日志文件大小超限告警

系统

- 新增 Querier 模块，提供 SQL 查询能力
 - 通过对 ClickHouse 数据进行抽象，向上提供 SQL 查询能力
 - 支持从页面上查看应用调用、网络流日志的数据表字段详情
- 采集器支持提取 Dubbo 头部中的字段作为 TraceID
- 采集器支持提取 HTTP 头部中的 X-Request-ID 字段
- 支持限制采集器运行环境的系统空闲内存比例
- 支持限制采集器的日志文件大小

v6.0.1

应用、网络

- 监控数据关联：统一的右滑页面展示任意服务的关联监控数据
 - 网络监控数据：指标、链路追踪、流日志
 - 应用监控数据：指标、流日志
 - 调用关系数据
 - 资源变更事件数据
 - 知识图谱数据
- 网络链路追踪
 - 支持 NFV 云网关全链路追踪
- 应用数据增强

- 应用流日志新增 Dubbo、Kafka 支持
- 易用性提升
 - 菜单切换时自动记忆搜索条件
 - 支持导出分发策略、流量下载策略
 - 指标曲线页面支持快速批量添加曲线图
 - 自动为推荐的指标量设置阈值

基础功能

视图

- 支持全局设置曲线图的补值方式

告警

- 新建告警策略时支持直接创建推送端点
- 新增推送端点测试功能
- 新增推送内容预览功能
- 曲线图无数据时支持触发告警

系统功能

资源

N/A

系统

- 支持腾讯 TCE 中的 NFV 云网关链路追踪
- 支持使用手机号码登录
- V6 开始修改为对监控资源数量和采集数据类型进行软件授权
- 系统组件日志全部存入 Loki

v6.0.0

应用

- 零侵扰采集
 - 新增支持解析 MySQL、Redis 流量，生成指标、链路追踪、访问日志数据并与资源知识图谱关联
 - 为应用层流量日志增加统计位置标记，便于定位微服务访问路径上的容器节点、宿主机、NFV 网关等关键网络位置
 - 支持基于 HTTP 流量日志数据绘制拓扑图，展现基于 HTTP 协议的微服务调用关系
- 搜索条件增强
 - 支持正则表达式过滤

- 支持按云平台资源名称、资源 ID 过滤资源
 - 支持按负载均衡监听器过滤资源
 - 支持为流量日志中的属性字段定义模板变量
 - 支持动态过滤所有租户、所有云平台
- **NFV 流量路径**
 - 支持展示腾讯专有云（TCE）中 NFV 区域的流量路径
 - 优化青云私有云环境下 NAT 网关、SLB 流量路径的绘制
- **子视图优化**
 - 流量拓扑图增加指标量差量表格，用于快速判断时延和丢包的瓶颈路径
 - 支持为 Top 折线图设置滑窗大小
- **易用性优化**
 - 流量日志数据增加指标量过滤能力、指标量字段支持排序
 - 优化链路追踪详情图的展示，便于快速定位性能瓶颈
 - 快速搜索框增加分组条件切换能力
 - 优化子视图的指标量切换操作
- **知识图谱**
 - 当云服务器作为 NAT 网关、负载均衡器载体资源时，支持在知识图谱中进行关联展示
 - 优化知识图谱 API 性能

网络

- 流量下载、流量分发策略不受容器 Deployment 临时删除影响
- 流量下载页面无数据时提醒用户切换时间范围

基础功能

视图

- 降低视图页面内存开销，优化页面预加载逻辑
- 缩减视图导出时的 JSON 文件大小

告警

- 告警事件推送消息模板中增加触发值变量
- 系统告警策略支持修改阈值
- 优化告警策略创建流程

系统功能

资源

- 同步云平台资源 ID
- 监控云平台 API 的返回数据指标

- 优化云平台告警展示，区分异常和警告
- 手动添加容器集群后立即生成关联的可用区和 VPC，简化 SaaS 租户上线流程

系统

- 采集器
 - 控制器仅向采集器同步可能存在访问关系的资源信息
 - 支持采集容器节点和云服务器上 lo 接口的流量
 - 采集器支持同时进行多种隧道协议解封装
 - 支持远程更新采集器配置文件
 - 支持远程控制采集器是否进行时钟同步、是否计算网络性能指标
- 控制器
 - 优化系统监控 API 性能
 - 支持使用腾讯 TCE 运营端账户登录

v5.7.3

全景图

- 计算能力增强
 - 按容器服务分组：支持将同一个容器服务的工作负载、容器 Cluster IP、容器 Node IP:Port 进行聚合
 - 调用关系自动展现：优化搜索条件语义，默认自动展现搜索资源与其他资源之间的调用关系
 - 流量日志增加 Avg 算子
- 数据增强
 - 流量日志增加采集网卡标识字段
 - 链路追踪中支持展示多个采集网卡名称
- 子视图增强
 - 支持切换查看流量日志详情中的精确指标值
 - 排名、详情类子视图支持跳转至链路追踪图
 - 仅使用路径数据绘制流量拓扑，以简化流量拓扑展示逻辑
 - 折线图的补值方式、表格的列宽分配方式支持记忆
- 易用性增强
 - 快速搜索框：新增快速搜索输入框，支持资源联想，简化搜索操作
 - 预置常用指标量组合，支持批量切换指标量
 - 高级搜索的分组条件和过滤条件中增加提示文案
 - 优化多选下拉框过滤搜索使用体验
 - 支持 PCAP 文件单独直接下载，支持 PCAP 文件下载时重命名

基础功能

视图

- Grafana Datasource 增强
 - 支持在 Grafana 中展示 DeepFlow 调用关系图
 - 支持在 Grafana 中使用容器集群的模板变量
 - 优化 Grafana 折线图图例

告警

- 支持查看告警事件的推送结果
- 系统告警事件中支持显示触发告警时的指标值
- 流量日志发送或写入采样时触发队列丢包的系统告警
- HTTP、Syslog 推送端点新增可读时间变量

系统功能

资源

- 解耦同步容器资源与云平台资源的循环，提升容器资源同步实时性
- 支持同步百度公有云应用型负载均衡、云数据库 RDS
- 支持同步青云私有云中云管区的物理机、虚拟机信息
- 支持同时对接同一个公有云平台上不同租户账号的资源

系统

- 采集器
 - 新增 ipvlan、weave CNI 支持
 - 支持提取 Skywalking v8 的 TraceID
 - 支持通过域名请求控制器
 - 降低采集器运行权限依赖，容器采集器支持以非 root 用户运行
 - 提升长连接流量中客户端、服务端方向判断的准确率
 - 专属服务器采集器支持解析采集点为虚拟网络的流量
 - 支持解析 VMware vSphere 的 ERSPAN 封装
- 控制器
 - 优化系统监控 API 性能
 - 支持将容器-P、Workload-V 类型采集器自动更新为容器-V 类型
 - 支持配置采集器自动发现时默认消耗的采集器授权类型
- 云原生：DeepFlow 部署方式使用 K8s 完全替代 Saltstack
- 计费：费用中心通知邮件文案优化

v5.7.2

全景图

- 计算能力增强
 - **Apdex**（应用性能指数）算子：时延类指标量支持 **Apdex** 算子，用于衡量被监控服务的用户满意度
 - 流量日志聚合：支持直接基于原始流日志、HTTP 日志、DNS 日志数据按资源和路径聚合，展示排名、分布、详情等子视图
 - 近似算子：新增 **Percentile** 和 **Distinct** 的近似算子，提升计算速度、降低资源开销
 - 策略匹配：**PCAP** 策略与分发策略支持按采集器过滤
- 数据增强
 - **Skywalking**：HTTP 日志中支持解析 **Skywalking v8** 的 **TraceID** 字段
 - 全栈链路追踪：流量拓扑中支持直接展示经过路由器、NAT 载体虚拟机、SLB 载体虚拟机的路径
- 子视图增强
 - **Grafana**：子视图添加到 **Grafana** 时支持 **Variables**（模板变量）
 - **Grafana**：支持将资源变更事件添加到 **Grafana**
 - **Histogram**（概率分布图）支持框选，快速过滤异常指标范围
 - 表格支持按内容自动分配列宽
- 易用性增强
 - 全栈链路追踪：直接快速查询资源关联的 **NAT** 网关、负载均衡器、容器服务等信息，实现一键搜索全链路路径
 - 知识图谱图形化展示效果优化
 - 高级搜索资源过滤时支持按 **IP** 反向选择资源
 - 流量搜索页面的默认配置支持重置
 - 流量搜索页面之间跳转时携带指标量，避免重新选择指标
 - 流量搜索页面支持切换至深色风格
 - 流日志详情页显示的时延、持续时间信息优化单位和进制显示
 - 优化错误信息的 **Tip** 展示，支持从 **Tip** 中复制 **IP** 信息
 - 优化子视图的查询性能

基础功能

视图

- 子视图支持直接跳转到全景图的分组切换、流量曲线、日志详情等页面

告警

- 自动化运维：告警事件的 **HTTP** 推送端点支持使用 **Jinja** 模板语法渲染推送内容

- 自动化运维：增加 Syslog 推送端点
- 告警策略列表中推送端点文案优化、Tip 优化
- 增加对采集器进程数、线程数达到限制的告警监控

系统功能

资源

- 腾讯 TCE：同步宿主机时标记腾讯 TCE 的 Underlay CVM 母机和 Overlay CVM 母机角色
- 青云私有云：支持同步青云负载均衡器 VIP
- 青云私有云：支持同步 DVR 模式私有网络
- 优化 OpenStack、公有云的资源信息同步
- 云服务器、安全组等列表中，支持查看全局的附属信息列表，例如查询所有安全组的规则
- 优化云平台录入时的区域和资源同步控制器填写方式
- 增加云平台同步的 API 调试工具，快速判断云 API 可能存在的问题

系统

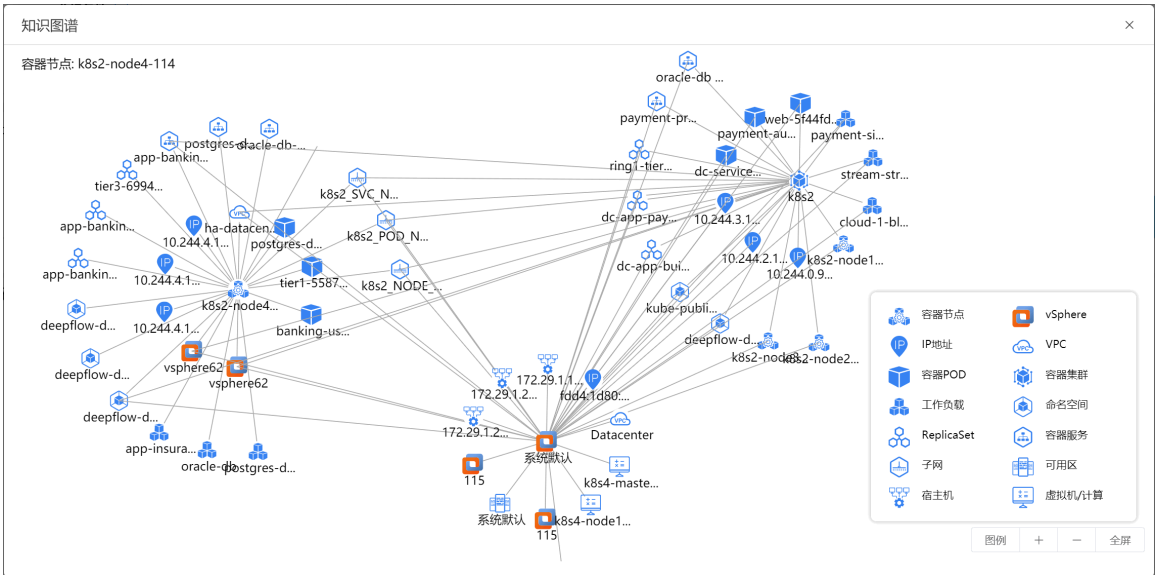
- 采集器
 - 安全性：采集器组配置支持限制进程数和线程数
 - 性能：支持配置流量过滤规则，忽略不关心的流量，适用于网关等高负载采集位置，降低采集压力
 - 腾讯 TCE：支持监控 TCE 实体网络
 - 支持采集器监控 DeepFlow 控制器和数据节点
 - 采集器性能监控数据支持按需存储
 - 支持容器 POD 跨集群 MAC 重叠
- 数据节点
 - 云原生：数据节点支持使用对象存储扩展存储能力
 - 数据节点磁盘消耗达到阈值时自动删除最旧的数据
 - 数据源文案优化
- 云原生：DeepFlow 容器化部署能力迭代
- 支持 KVM 采集器的控制 MAC 变更
- 取消租户账号查看宿主机资源的权限
- 控制器页面列表展示性能优化

v5.7.1

全景图

- 流量日志支持搜索所有属性字段
 - 流日志：Internet IP 等

- HTTP 日志：请求方法、代理客户端、主机名、路径、请求 ID、TraceID、响应码等
- DNS 日志：请求 ID、查询域名、查询类型、响应码、解析地址等
- 图形化展示资源知识图谱



- 链路追踪支持关联采集网卡名称



- 数据增强
 - 流量下载功能支持保留原始包头字段（不压缩）
- 计算增强
 - 高级搜索条件增加统计位置
 - IP 地址类型的模板变量支持直接输入以切换当前取值
 - 模板变量取值为全选时子视图支持点击跳转切换搜索条件

- 子视图增强
 - 流量日志详情页新增查看 API 操作
 - 表格中的空值在排序时始终处于末尾
- 易用性改进
 - 高级搜索增加快速复制能力
 - 跳转到流量曲线页面时自动展示上级页面选择的指标量
- UI 优化
 - 表格总高度支持自适应调整
 - 优化子视图无数据时的显示内容
 - 优化页面中错误信息的显示位置，优先显示在子视图内部
 - 页面展示指标量的说明信息
 - 表头筛选按钮显示优化

基础功能

视图

- 视图删除流程简化
- 深色模式 UI 优化

告警

- 部分系统告警聚合开始和结束事件，降低告警事件数量
- 支持修改系统告警的等级

系统功能

资源

- 适配性提升
 - 支持同步百度公有云资源
 - 支持自动同步腾讯 TCE 中的所有 TKE 容器集群
 - 容器资源同步取消对 ConfigMap 权限的依赖
- 易用性改进
 - 支持同步云平台附属容器集群的整体删除事件
 - 附属容器集群列表支持模糊搜索
- UI 优化
 - 优化展示附属容器集群的汇总错误信息
 - 统一资源列表中的查看详情操作行为

系统

- 数据节点存储引擎切换至 ClickHouse，实现流量日志全量（无采样）存储
- 采集器远程 Debug/Profile 端口合并至 UDP 20035，且无需采集器主动监听

- 易用性改进
 - 新增采集网卡列表
- 租户账号
 - 限制租户账号对资源的增删改权限
 - 全景图搜索历史和视图支持共享给所有账号
- 授权与计费
 - 公有云场景下支持按服务类别、采集器规模、采集器运行时长的计费模式
 - 完善软件授权的邮件通告

v5.7.0

全景图

- 全栈链路追踪
 - 图形化展示混合云（虚拟化+容器、公有云+私有云）场景下的全栈链路追踪，图形化展示每条链路的指标数据增量占比
 - 覆盖从客户端到服务端，沿途容器节点、宿主机、网关宿主机、网关等 **11 个关键链路**，逐跳展示每个采集网口上的指标数据
 - 增强微软 ACS 全栈链路追踪能力，支持展示经由 MUX/Gateway 上云下云的全链路路径
 - 增强青云 QingCloud+QKE 混业云全栈链路追踪能力，支持智能网卡 VXLAN Offloading、LXC SLB 场景
 - 支持监控虚拟网络流量 Flood 在所有宿主机、容器节点上的传播路径
 - 增加支持对 Calico IPIN 隧道的解析能力
- 模板变量
 - 全景图、视图的搜索条件支持填写模板变量，实现搜索条件的快速切换
 - 模板变量可随全景图搜索历史、视图分享，资源可视范围可随分享的目标租户账号自动调整
 - 支持按模板变量的不同取值分别记忆流量拓扑图的排布位置
- 知识图谱
 - 支持 VPC 中内网 CIDR 与云平台全局 CIDR 部分重叠的场景
 - 支持关联展示 KVM 宿主机、容器节点上 API 未返回的额外路由 IP
- 数据增强
 - 原始指标数据新增时延峰值，记录 1 秒内时延最大值，用于发现偶发时延抖动
 - HTTP 日志与 DNS 日志中同一个会话的请求和响应合并为一条记录
 - 流量下载页面支持创建虚拟网络全量 Payload 存储的 PCAP 策略
 - 流量下载页面支持创建容器资源的 PCAP 策略
 - 流量下载页面支持多租户查看

- 计算增强
 - 高级搜索支持按资源类型过滤
 - 高级搜索自动分组新增将容器 POD 聚合为工作负载的能力
 - 高级搜索增强通过宿主机、容器节点、负载均衡器、容器 Ingress、容器服务快速过滤其关联资源的能力
 - 支持随指标数据同时展示各类型资源变更事件数量
- 子视图优化
 - Top 折线图支持点击查看资源知识图谱信息
 - CSV 下载时自动增加易读形式的时间列
 - 查看 API 功能支持直接复制整个 API URL 和请求体
 - 支持调整并记忆折线图表格形式指标量显示高度
 - 排名类子视图支持按升序显示
 - 支持子视图 Y 轴值域范围切换为相对范围，在指标数据值域范围小时便于直观展示数据细微抖动
- 易用性改进
 - 时间控件中支持直接填写相对时间
 - 支持全局切换页面中所有子视图的数据源
 - 页面跳转
 - 支持快速切换展示资源的客户端、服务端流量访问关系
 - 所有链接跳转以新页面方式打开
 - 支持直接访问流量曲线页面
 - 优化切换资源分组跳转时自动携带的搜索条件
 - 高级搜索
 - 高级搜索中的资源过滤支持表格形式，解决资源同名时的选择问题
 - 支持忽略高级搜索中的指定条件
 - 文案及视觉
 - 改进时延指标量的单位显示
 - 改进子视图 Loading 效果
 - 设置指标量弹出框紧凑设计
 - 改进表格的列宽分配，复选框列和操作列紧凑排布
 - 优化搜索历史的排序和星标、分享展示
 - 优化大规模下拉框的下拉选择，将选中内容在下拉框中的顺序固定

包分发

- 分发策略支持选择 CIDR 动态变化的子网，当 CIDR 变化时策略自动跟随
- 分发策略和 PCAP 策略支持选择多个资源，自动生成资源组

- 自动生成容器节点、容器 POD、容器服务所在子网，以简化容器分发策略的配置
- 优化分发策略和 PCAP 策略新建页面排版

基础功能

视图

- 模块化折叠能力
 - 支持在视图中创建子模块容纳多个子视图
 - 支持折叠子模块以加速页面加载
- 支持全局切换页面中所有子视图的数据源
- 流量拓扑图添加到视图中时携带资源节点的排布位置
- 时间控件增加页面刷新快捷操作

告警

- 告警事件推送端点
 - 增加 HTTP 推送端点，可将告警事件推送至外部告警平台或工作流系统
 - 增加 PCAP 推送端点，可通过告警事件触发 PCAP 策略的启用、禁用
- 增强系统告警能力
 - 云平台同步异常的时发送告警
 - 资源删除导致分发策略删除时发送告警
 - 利用 eBPF 支持监控系统进程的快速/高频/持续重启
- 易用性改进
 - 告警策略列表支持显示告警事件数量
 - 优化系统告警，显示进程重启的名称

报表

- 报表中的子视图支持下载 CSV 数据

系统功能

资源

- 自动同步
 - 腾讯 TCE：增加支持同步 SLB（负载均衡器）及负载均衡规则
 - 青云 QingCloud：增加支持同步基础网络和未加入 VPC 的网络
 - 微软 ACS：增加支持同步 MUX 虚拟机、MUX 宿主机
 - 5G 核心网/5GC：增加支持容器 POD 多网卡、多 IP 场景
 - Kubernetes：增加支持附属容器集群跨越多个可用区的场景
 - OpenStack：增加支持未部署 L3 Agent 时同步路由器信息

- 支持同步宿主机、容器节点的 CPU 和内存信息，用于指导采集器资源限制的配置
- 支持为所有云平台附加容器集群，关联分析混合云全栈链路追踪
- 支持无法调用云平台 API 时通过容器采集器同步所在虚拟机的资源信息
- 手动录入
 - 支持从页面创建对等连接
 - 虚拟机修改为云服务器，区分 VM 和 BM，区分计算、网络、存储类型
 - 支持从页面手动录入云服务器及其网卡、宿主机及其网卡
- 性能优化
 - 优化采集器对运行环境网卡信息同步的性能，降低传输带宽消耗
 - 支持设置云平台同步的区域白名单，加快公有云资源同步速度
- 易用性改进
 - 表格中的名称列和 IP 列支持一键复制
 - 所有云平台均支持从页面录入 API 对接信息，支持修改云平台名称
 - 优化容器集群列表展示，关联云平台和 VPC
 - 优化 IP 地址列表展示，关联 MAC 地址，若同时属于容器节点和云服务器时展示为两条
 - 所有资源列表支持列选择

系统

- 账号权限
 - 支持对接外部 Radius、Windows AD（OpenLDAP）账号系统
 - 全景图搜索历史、视图支持共享给所有管理员
- 授权管理
 - 支持 C 类采集器授权，用于虚拟化+容器全栈混合云环境
 - 页面展示 License 授权量的消耗情况
- 部署优化
 - 支持通过采集器同步容器资源信息，DeepFlow 控制器无需主动请求 K8s Master 节点
 - 一键完成部署流程中的操作系统初始化配置
 - 支持远程批量部署 Windows 环境采集器
 - 支持自动发现 Workload-P、专属服务器类型采集器
 - 支持完全关闭采集器请求控制器和数据节点的负载均衡机制，依靠云平台 SLB 调度
 - 精简容器采集器运行所需权限
 - 支持混合云环境中不同采集器使用相同控制 IP
 - 部署命令行完整参数支持反向查看

- 运维优化
 - 支持在 DeepFlow 控制器上远程批量升级容器采集器
 - 支持采集器自动发现时自动加入指定的分组，避免加入默认组导致无法正常工作
 - 支持从页面配置 Hyper-V 类型采集器的采集网口
 - 当云服务器安装为容器节点时，支持 Workload 和容器类型的采集器自动更新
 - 优化多区域部署的运维能力，CLI 命令自动判断所属区域是否为主区域
 - 支持一键升级 DeepFlow Patch
- 性能优化
 - 控制器性能优化，策略下发进程 CPU 消耗降低 30%，提升页面的加载速度
 - 数据节点时序数据读写性能优化，写入性能提升 80%~300%，读取性能提升 30%
 - 采集器支持 BPF 动态自适应调整流量采集包长，提升采集性能
 - 多区域部署下 DeepFlow 自身性能监控数据仅存储于本区域，降低跨区域带宽消耗
- 易用性改进
 - 优化页面的采集器配置显示，高亮显示相对于默认配置或全局配置的修改内容
 - 支持在列表表头中通过选定枚举值过滤表格内容
 - 告警事件、报表下载、系统日志、采集器运行日志、系统进程日志、云平台同步时间页面的时间控件独立，不影响全景图的时间记忆
 - 支持输入 IPv6 地址的完整形式或压缩形式匹配同一个 IPv6 地址

v5.6.3

全景图

- 知识图谱
 - 资源分组、资源知识图谱、资源过滤候选项从 **16 项增加到 28 项**
 - 新增资源池属性，包括云平台、租户
 - 新增网络资源属性，包括路由器、DHCP 网关
 - 新增网络服务属性，包括安全组、NAT 网关及其资源关联、负载均衡器及其资源关联
 - 新增存储资源属性，包括 RDS、Redis
 - 新增/增强容器资源属性，包括集群、节点、服务及其资源关联、Ingress 及其资源关联、Ingress 策略
 - 新增**自动分组**能力，IP 地址自动聚合为最精细的计算、网络、存储、容器资源

- 流量拓扑支持展现资源之间基于容器服务、负载均衡监听器、NAT 网关策略的 **DNAT 路径关系**，并支持手动补充关系线
- 资源变更事件支持展现容器 **StatefulSet POD 的迁移、重建**，并支持展示容器 POD 的 IP 变更事件
- 链路追踪
 - 支持 **IaaS+PaaS 全栈混合云**场景下的虚拟网络访问路径链路追踪
 - 覆盖从客户端到服务端，沿途容器节点、宿主机、网关宿主机、网关共 **10 个虚拟网络关键链路**
- 应用感知
 - 新增 HTTP 日志和 DNS 日志功能，可关联查看指标数据的所有 HTTP 和 DNS 访问日志
 - HTTP 日志支持提取 **APM TraceID**，支持 **Datadog、SkyWalking、Zipkin、Jaeger** 等 APM，实现与 APM 工具的联动
 - HTTP 日志支持提取 X-Forwarded-For 字段，以展现负载均衡真实源 IP
- 告警能力
 - 新增绝对跨度、相对跨度、标准差 3 个算子，以支持对**陡增陡降、毛刺**进行告警
 - 指标量从 12 类 108 种 1102 个增加到 **12 类 114 种 1810 个**
 - 新增 Top 折线图告警设置能力，实现**大批量云资源一键设置**固定阈值告警
 - 新增**基线阈值**展现能力，支持 **LSTM 和 ARIMA** 两种算法计算折线图的动态基线
 - 支持为子视图中选择的所有指标量分别设置阈值，Tip 中所有超过阈值的指标量标红
- 日志数据
 - 流日志支持分别记录双向最外层隧道 IP 信息
 - 流日志支持按日志 ID、客户端口过滤查询
 - 支持分别下载流量在所有采集网口的 PCAP 文件
- 易用性改进
 - 由于部分数据节点故障仅返回部分数据时，页面显式提示用户
 - 所有子视图
 - 所有标准操作仅当鼠标悬停时显示，同时优化操作名称使得体现当前子视图状态
 - 流量拓扑、流量详情表格、流量排名 Tip 支持显示资源图例，优化访问地图图例
 - 子视图的图例名称支持切换为全名显示方式
 - 页面支持自定义子视图的颜色、边距、局部显示隐藏等 UI 样式
 - 子视图支持弹框展示

- 流量拓扑
 - 优化细节 UI 效果，物理拓扑区域高度可调整，拓扑与表格高亮联动
 - 抑制流量拓扑中展示的总资源数量，清除流量拓扑中的孤立点
 - 支持展示 **IaaS+PaaS 全栈混合云**场景下访问路径的链路追踪数据
- 趋势类子视图：支持选择三种补齐空值的方式，同时优化折线图的阈值线显示
- 表格类子视图：支持列宽重置，支持高亮表格过滤内容

包分发

- 支持配置物理网络分光镜像流量的分发策略

基础功能

视图

- 支持视图导出导入
- 子视图支持直接编辑搜索条件
- 视图、子视图支持另存
- 简化告警策略设置，与指标量的阈值设置联动

告警

- 告警事件中支持直接展示对应的时间序列图

系统功能

资源

- **IaaS+PaaS 全栈混合云**场景下，支持关联容器节点和虚拟机
- 新增支持同步青云 QingCloud+QKE、青云 KubeSphere、金山银河云资源
- 完善页面上可直接添加的云平台类型，增加对腾讯专有云、青云公有云、微软公有云的支持
- 支持为子网指定 VIP 属性，用于依靠 MAC 地址做路由的负载均衡场景，例如微软私有云 MUX 等
- 增加 DHCP 列表，支持同步 ReplicaController，完善资源与可用区的关联关系

系统

- 支持租户
 - 支持手动创建租户账号，并赋予租户关联的 **VPC 或容器命名空间**资源
 - 所有账号之间支持以只读或读写的方式**共享全景图搜索历史、视图**
 - 租户可使用全景图流量搜索、视图、告警、报表、资源功能

- 全景图、资源页面支持通过租户过滤资源，全景图知识图谱支持显示租户信息
- 采集器
 - 适配性增强
 - 采集器通信端口大幅降低，仅需请求控制器和数据节点的各一个端口
 - 支持页面配置隧道解封装协议，目前支持解析 **VXLAN**、**双层 VXLAN**、**TCE GRE** 隧道封装
 - 容器节点无法放开 **privileged** 权限时，自动切换为通过 ARP 流量推算 POD 的实际 MAC 地址
 - 支持通过虚拟机的 QEMU XML 文件获取 KVM 虚拟机的实际 MAC 地址
 - 支持多区域之间 MAC 地址重复场景下的资源知识图谱信息标记
 - 隧道解封装采集器适配所有网卡，自适应选择高性能或普通收包模式
 - 易用性增强
 - 页面展示采集器的**配置信息和运行日志**
 - 优化采集器批量操作，优化采集器加入组操作
 - 采集器列表支持 CSV 下载
 - 配置能力增强
 - 新增 **HTTP 日志的代理客户端**、**TraceID 字段名**配置项
 - 新增日志发送速率、日志打印等级、日志存储时长配置项
 - 新增应用层日志解析包长、流日志采集速率、应用层日志采集速率配置项
 - 新增应用层指标数据、包统计数据类型功能开关
 - 新增过滤流日志、过滤应用层日志配置项
- 运维能力
 - 新增支持**通过 Web 页面向控制器上传、下载文件**
 - 新增轻量级 **Web 端 SSH 登录页面**，不再依赖单独提供 SSH 运维方式
 - 支持显示系统组件的运行环境 CPU、内存、体系架构、操作系统、内核信息
 - 支持自动清理 ES 中的过期系统日志、操作日志、告警事件、资源变更事件、**APM 监控数据**
- 部署增强
 - 支持多区域独立部署，减少区域之间通信端口号
 - 支持通过从控制器同步云平台、容器集群资源信息
 - 完善容器化部署
- 性能优化

- 优化专属服务器采集器流量采集性能
- 优化数据节点查询性能

v5.6.2

全景图

迭代功能聚焦在增强对**容器**的支持，增强**混合云异构资源**的流量检索能力，增强**网络性能**、**安全态势**监控能力，关联指标数据与**流日志**数据，增强页面**自定义**能力，优化**易用性**和**流量拓扑**展示。

- 计算和数据增强
 - 指标量从 8 类 41 种 471 个增加到 **12 类 108 种 1102 个**
 - 增加重传比例、零窗比例、建连失败比例、HTTP/DNS 异常比例等比例指标量，用于直观展现异常比例
 - 增加传输层负载（并发连接数）指标量，用于诊断 NAT/LB 网元负载
 - 增加传输层 TCP 超时、应用层 HTTP 超时、DNS 超时，用于监控无响应场景
 - 增加网络层有效载荷指标量，用于诊断网络丢包
 - 增加平均包长指标量，用于监控网络传输效率
 - 增加资源、路径及流量属性的基数指标量，用于监控主机扫描、端口扫描等安全态势
 - 包属性维度从 2 种增加到 **4 种**
 - 增加 TTL 统计维度，用于诊断网络层环路
 - 增加包长区间统计维度，用于监控包长特征
 - 算子从 6 种增加到 **8 种**
 - 增加 Percentile 算子，用于计算百分位数
 - 增加 Distinct 算子，用于计算资源属性基数，例如访问的 IP 数、被请求的服务端口数等
 - 资源知识图谱维度从 15 种增加到 **16 种**
 - 增加资源关联的容器 ReplicaSet 信息，用于监控容器灰度升级过程
 - 折线图增加滑动窗口大小设置能力，实现对时间序列的平滑处理，抑制告警事件
 - 流日志并入全景图流量搜索，实现流量指标数据与五元组流日志数据关联
 - PCAP 下载并入全景图，并新增支持配置含 Payload 的虚拟网络流量策略
- 子视图增强

- 趋势类子视图
 - 支持 Tip 同步显示
 - 支持显示所有指标量
 - 支持多条折线堆叠显示
 - 支持显示表格形式的统计值摘要
- 拓扑类子视图
 - 流量拓扑支持展示资源监控数据表格
 - 流量拓扑根据业务画像服务依赖补齐资源连线
 - 流量拓扑优化节点排布
 - 流量拓扑支持动态显示合适的资源图标
 - 流量拓扑中物理拓扑部分支持整体拖动
- 表格支持列宽调整，支持选择显示列
- 地理分布图支持设置指标量阈值
- 所有子视图支持全屏显示
- 易用性改进
 - 高级搜索支持多个资源集合、支持 NOT 语义过滤、支持多组 OR 语义的资源过滤条件，更好的呈现异构资源下的云原生业务访问关系
 - 页面折叠区域的尺寸、顺序及子视图数量支持自定义
 - 改进时间选择控件，增加最近搜索历史
 - CSV 文件列名、列序改善
 - 改进返回上级页面的体验

包分发

- 分发策略支持过滤所有网络层协议
- 分发策略对端过滤规则支持仅填写端口号
- 分发策略允许修改分发点和标签

基础功能

视图/告警/报表

- 子视图支持点击弹出全景图资源知识图谱信息
- 告警策略与折线图阈值设置联动
- 支持为视图中的流量拓扑图生成报表

资源

- 通过阿里专有云 - Apsara Stack 兼容性测试互认证，成为阿里云首个混合云网络流量分析合作伙伴
- 通过腾讯专有云 - TCE (Tencent Cloud Enterprise) 兼容性测试互认证，成为腾讯云首个混合云网络流量分析合作伙伴

- 支持同步容器的 ReplicaSet 信息，并与全景图联动
- 支持管理员在页面上配置云平台对接信息，支持查看云平台同步的健康状态
- 增加区域、可用区、云平台的品牌图标展示，并与全景图联动
- 支持微软公有云资源同步

系统

- 使用 OpenTracing 监控全景图 API 性能
- 支持解析腾讯 TCE Underlay 隧道封装流量
- 采集器
 - 支持采集器自动注册
 - 支持在页面上修改所有采集器配置信息
 - 支持在页面上展示采集器的版本信息，采集器列表支持选择显示列并随账号记忆
 - 采集器远程升级时检查二进制文件的可用性
 - 采集器远程日志和自身性能监控数据统一请求控制器的 20033 端口，减少端口使用
 - Linux 环境下采集器关联的数据节点变更时，采集器无需再重启
- 部署
 - 支持在 K8s 中云原生部署所有 DeepFlow 组件
 - 支持控制器或数据节点被 NAT 网关或 SLB 代理的环境下部署
 - 支持自动识别容器采集器运行虚拟、物理环境
 - 授权证书支持从 Workload 类型采集器升级为容器类型采集器

v5.6.1

包分发

- 分发策略采集点过滤规则支持容器服务
- 分发策略支持配置 Headless 类型的容器服务

全景图

迭代功能聚焦在增强**容器支持**、增加**L7 指标量**、增加**资源变更事件关联展示**、增强**数据源的选择和自定义能力**、增强**流量搜索易用性**和**流量拓扑展示**。

- 计算和数据增强
 - 增加 **1 种**数据源
 - 增加包统计双端数据
 - 支持展示包统计流量拓扑，支持展示组播流量拓扑
 - 数据源时间粒度从 2 种增加到 **4 种**
 - 新增小时、天粒度的自定义数据源

- 所有子视图支持切换数据源
- 新增 **5 种**资源变更事件
 - 支持展示虚拟机的创建（同步）、删除、启停、迁移、IP 增删改等变更事件，与流量监控数据关联展示
 - 支持展示容器 POD 的创建（同步）、删除、IP 增删改等变更事件，与流量监控数据关联展示
- 流量聚合维度从 **11 种**增加到 **13 种**
 - 增加容器命名空间、容器工作负载的过滤条件和分组条件
- 资源知识图谱维度从 **11 种**增加到 **15 种**
 - 增加资源关联的容器命名空间、容器工作负载、资源组服务、容器服务知识图谱信息
- 指标量从 **4 类 28 种 107 个**增加到 **8 类 41 种 471 个**
 - 增加 UDP 应用时延，增加 TCP 建连异常
 - 增加 HTTP 和 DNS 吞吐量、时延、异常指标
 - 支持计算指标量的百分比数值
 - 流量拓扑中支持展示端到端路径、物理链路、物理网元引入的指标量差量，以体现丢包和时延
 - 优化 TCP 建连时延计算方法以体现重传对建连的影响
- 子视图增强：从 **9 种**增加到 **12 种**
 - 增加 Top N 折线图，便于用户比较 Top 分量的指标历史曲线
 - 增加趋势分析柱状图，便于用户定位时间范围
 - 增强表格，支持选择展示的列
 - 增强折线图，支持序列选择，支持时间范围选择
 - 增强流量拓扑图
 - 支持虚拟网络和物理网络同时展示
 - 支持资源和流量属性同时展示
 - 支持在关联表格中展示拓扑图的路径数据，并支持过滤及与拓扑联动
 - 增加自动化节点位置排布功能
 - 支持缺失数据补齐功能，展示 Top N 以外的数据便于计算差量
- 易用性改进
 - 折线图页面增加自定义 Tab 页
 - 支持记忆流量搜索页面的 Top 选择、指标量选择等临时配置
 - 支持保存搜索条件、时间范围、页面临时配置
 - 搜索结果中支持直接展示资源的知识图谱信息
 - 优化流量搜索页面排布，在一个页面中通过折叠的方式展示更多信息
 - 优化高级搜索框，优化搜索条件修改方式

- 播送类型的统计结果中展现采集器信息，便于用户定位 Flood 发生的位置
- 网络拓扑
 - 支持云企业网路由连通性检查

流回溯

- 易用性改进：切换采集点不再清空搜索条件

基础功能

视图

- 全景图中新增的子视图、算子选择、数据源选择支持添加至视图

告警

- 支持为告警策略添加自定义标签
- 易用性改进：优化告警事件列表展示

报表

- 全景图中新增的子视图、算子选择、数据源选择支持在视图中生成报表

资源

- 支持同步阿里公有云 RDS 和 Redis 资源
- 支持同步容器节点 IP 地址、节点 POD 网段、节点 CNI 插件 IP 地址
- 区域、可用区移动至资源池菜单，并新增云平台列表
- VPC 和子网列表中增加 POD 数量展示
- 业务画像并入资源菜单
- 业务画像创建容器资源组时支持指定集群和命名空间过滤

系统

- 支持添加多个授权文件
- 授权文件支持 A 类采集器和 B 类采集器中的授权共享
- 支持配置和生成全景图数据源
- 支持使用系统 UDP 发送压缩包头和分发流量，避免 ARP 请求失败的影响
- 数据节点与从控制器和专属服务器采集器解耦
- 易用性改进：去除页面左边栏，为页面提供更大可视空间

v5.6.0

包分发

- 分发策略
 - 支持分发容器 Service 流量
 - 支持关闭全局去重功能（需要修改后台配置文件）

- 流量监控
 - 支持查看分发策略匹配的流量统计值
 - 支持查看分发点匹配的流量统计值
- 隧道封装
 - 支持在原始包中加入 VLAN 头，以避免后端 TAP 交换机、流量汇聚设备的复杂配置（需要修改后台配置文件）
 - 支持在隧道头、原始包 VLAN 头中标记流量采集的位置（源端、目的端），也可视为相对分发策略中采集点的方向，可用于容器环境下标识服务端 IP
 - 支持利用 ERSPAN 头部 48 bit 记录网包采集的时间戳，可表达的精度约为 15 微秒
 - 支持将流量标签填写在 GRE 头的 Key 字段中
 - 支持配置分发流量使用的 VXLAN 隧道端口

全景图

- 流量搜索
 - 支持十一个维度的资源流量搜索和资源知识图谱展现
 - 任意图表可添加至视图、设置指标量、查看 API、下载 CSV 数据
 - 所有图标支持的操作标准化，例如切换图表
 - 支持展示 29 类、107 种指标量统计值
 - 流的双端统计、资源统计支持所有指标量的秒粒度数据
 - 支持流量拓扑图（基础拓扑、地理拓扑）、柱状图、饼状图、表格、Top N 分布图、访问地图、物理拓扑、折线图等多种统计数据展现方式
- 网络拓扑
 - 连通性检测放开 IP 到 IP 条件
 - 连通性检测支持云企业网、DNAT
 - 增加物理拓扑及采集点器计算的会话表、ARP 表展现

流回溯

- 流日志
 - 支持配置采集点过滤条件，仅存储特定采集点的原始流（需要修改后台配置文件）
- PCAP
 - 支持查看分发点匹配的流量统计值

基础功能

视图

- 增加视图列表
- 视图增加报表功能

- 子视图支持标准化操作
- 时间维度折线图类型子视图支持设置告警和展示告警事件
- 折线图支持大范围数据区间选择及放大缩小能力

告警

- 告警策略支持子视图设置的阈值告警

报表

- 报表策略支持从视图中创建

业务画像

- 业务范围支持子网
- 资源组新增关键服务列表，标记资源组的关键服务端口
- 新增三种类型资源组：容器服务，容器服务后端工作负载，负载均衡器后端主机，可自动关联资源的变化
- 将业务拓扑和流量拓扑、服务发现合并展示

资源管理

- 支持手动录入子网以替代监控网段的功能，支持为子网设置区域、可用区、VPC 属性
- 新增图例列表，支持为资源组增加图例，支持修改物理网元的图例
- 新增支持 F5 负载均衡策略信息同步
- 已支持云平台新增资源同步
 - 增加可用区列表，若私有云无可用区概念时对应一个可用区，其他情况下同步云平台的可用区列表
 - 容器平台新增服务（Service）、Ingress 同步
 - 阿里云支持云企业网同步
 - 阿里云、AWS、腾讯云、OpenStack、华为云、平安云支持自动添加安全组默认策略
- 新增支持云平台
 - 华为公有云
 - 青云公有云
 - 青云私有云

系统管理

- 去掉分析器组件，替换为三种组件的集合：数据节点、从控制器、共享服务器类型的采集器
- 数据节点支持关联至一个区域的全部可用区，当区域内新增可用区时不再需要手动修改关联关系
- 去掉租户，使用多个管理员来满足多用户的需求。

- 采集器适配性提升
 - 采集器支持以 TCP 传输压缩包头
 - 采集器支持部署于纯 IPv6 环境中
 - 采集器支持 KVM 多 IP（采集器通信 IP 与 OpenStack 管理 IP 不一致）运行环境
 - 采集器支持运行于 K8S CNI 使用 Calico 的网络环境中
 - 采集器支持接收 NetFlow/NetStream v5、sFlow 数据
 - 采集器支持的物理采集点数量提升到 250 个
 - Workload-V 类型的采集器支持自动同步虚拟机资源信息
- 避免控制器、数据节点磁盘空间耗尽
 - 日志：检查 /var/log 目录，当剩余空间不足 10%或 5G 时，删除最老的日志文件
 - 报表：检查控制器报表存储目录，当剩余空间不足 10%或 5G 时，删除最老的日志文件
 - TSDB：检查数据节点 TSDB 数据目录，当剩余空间不足 10%或 5G 时，删除最老的 Shard
 - Corefile：限制每个进程最多生成一个 Core 文件

功能介绍

总览

总览是面向 DeepFlow 管理员的平台整体状态展现。

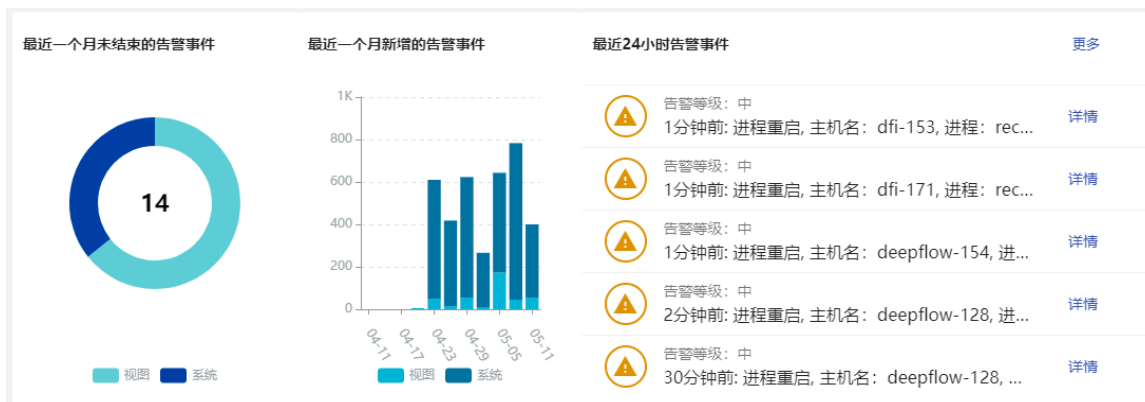
功能使用

总览页面呈现整个系统运行的概要信息，分为如下图三行。

展现监控网络中采集器、物理采集点、虚拟采集点数量，展现最近一小时采集和分发总流量。



展现最近一个月未结束的告警事件分布、新增的告警事件分布和趋势，及最近 24 小时的最后 5 条告警事件列表。



展现业务层面统计信息包括云租户、业务、VPC、外网 IP 地址数量，以及网络层面统计信息包括区域、可用区、宿主机、容器节点数量。



功能规格

授权

基础功能，无需授权。

限制

- 仅当与云平台同步后才可显示页面底部的外网 IP、宿主机数量
- 仅当与容器平台同步后才可显示页面底部的容器节点数量

应用/网络

全景图是 DeepFlow 两大功能之一，流量监控指标和网络知识图谱全景可视化，通过强大的搜索能力提供云网全景视图下的监控诊断解决方案，结合流日志和 PCAP 数据提供完整的故障回溯取证能力。

流量指标

功能简介

全景图流量搜索功能聚焦于从不同的维度对虚拟网络中的流量指标数据进行灵活检索。所有可视化图表（子视图）均可加入视图中进行进一步的组合、设置告警生成策略、设置报表生成策略。

术语解释

- 采集点：标记流量采集的位置，分为物理采集点和虚拟采集点
- 监控资源：监控资源=客户端、服务端时，表示对应的资源的在流量中的角色

- 统计位置：双端统计场景下，两个资源之间的流量统计值在客户端/服务端、客户端/服务端容器节点、客户端/服务端宿主机、客户端/服务端网关宿主机、客户端/服务端网关均能获取到，统计位置=客户端表明是在客户端侧获取到的流量的统计结果
- 指标量：DeepFlow 支持网络层至传输层（L4 至 L7）的吞吐、负载、时延、性能、异常，及资源、路径等属性共 12 类 110 种 1170 个指标量
- 算子：每种类型的指标量可以通过使用一系列算子进行统计计算，目前支持 Sum（总量）、Avg（均值）、Max（峰值）、Min（谷值）、Spread（绝对跨度）、RSpread（相对跨度）、StdDev（标准差）、Percentile（百分位数）、Distinct（基数），支持计算百分比和路径上的指标量差量
- 统计值：指标量在某个时刻的具体值即为统计值
- 数据源：DeepFlow 支持 1 秒、1 分钟两种内置数据源，用户可创建 1 小时、1 天的自定义数据源，数据源间隔表示同一个资源对象记录原始统计值的最小时间间隔

功能使用

功能页面

流量搜索从十余个维度的资源视角展示网络性能监控数据。

资源分组的十余个维度包括：- 自动分组：自动分组-服务聚合、自动分组-工作负载聚合、自动分组-全展开 - 计算资源：宿主机、云服务器 - 容器资源：容器集群、容器节点、命名空间、工作负载、容器 POD - 网络资源：VPC（Virtual Private Cloud）、子网、IP 地址 - 业务画像：业务、资源组（由业务画像生成） - 资源池：区域（Region）、可用区（AZ，Availability Zone）

通过点击拓扑中的节点（代表资源）、路径（代表资源之间的访问关系），以及分布图中的分组（代表资源或访问关系），用户可对当前的维度进行层层细化。

统计数据展示的视角（页面）包括：- 指标数据：数据以 IP、IP 二元组、服务端口聚合，在资源、路径统计数据基础上展示性能指标 - 流日志：数据以 IP 五元组聚合，在流日志数据基础上展示性能指标 - HTTP 日志：数据以 HTTP 请求、响应聚合，在 HTTP 访问日志基础上展示性能指标 - DNS 日志：数据以 DNS 请求、响应聚合，在 DNS 访问日志基础上展示性能指标

通过点击拓扑中的节点、路径，以及分布图中的分组，用户可在上述页面之间进行切换，从不同的视角对同样的数据进行展现。另外，上述页面还可跳转到**指标曲线**页面，以折线图的视角，展现现拓扑中的节点、路径或分布图中分组的统计数据在不同时间的结果。还可跳转至**资源变更事件**，与云服务器、容器 POD 的启停、创建（同步）、删除、迁移、IP 变更事件进行关联展示。且可查看选中节点或路径的**调用关系**。也可点击路径上的**链路追踪**，查看链路上的指标详情。进一步的，可以跳转到**日志详情**页面，查看对应的原始流量日志详细信息。

功能使用： - 时间选择：点击右上角时间控件，可以定位选定时间内的数据展示，开始时间、结束时间支持填写精确时间、相对时间，例如：2021-03-10 15:08:37、now、now/d、now-6d，相对时间支持的时间单位为：s (seconds)，m (minutes)，h (hours)，d (days)，w (weeks)，M (months)，Q (quarters)和 y (years) - Tip 同步开关：流量搜索、流量曲线增加 Tip 同步开启按钮，实现当鼠标移至某子视图（趋势分析、资源流量排名 Top 折线图、路径流量排名 Top 折线图、流量排名、资源变更事件及所有折线图）后，所有上述子视图 Tip 显示 - 搜索历史-分享：可将设置好的流搜索历史分享给单个账号、管理员账号集合、所有账号集合（单个管理员赋权优先于全部管理员/所有账号赋权），赋予其只读/读写权限，分享来的搜索历史在页面有橘色只读/读写标识 - 禁用/启用：用于临时禁用某个搜索条件 - 保存：按照以下四种方式保存 - 记忆时间范围：记忆时间控件中选择的时间范围、时间间隔 - 记忆拓扑位置：指记忆流量拓扑图的资源相对位置 - 记忆页面配置：指部分子视图的操作配置（补值方式、均分列宽、按内容分配列宽等）、指标量配置等 - 记忆模板变量取值：记忆模板变量及选中的值

搜索条件

通过输入不同的搜索条件，可实现对统计数据多种维度的过滤、分组、聚合。流量搜索的支持输入多个资源集合，当业务部署在异构资源池中时，可通过多个资源集合的组合将业务呈现在一个视图中。

通过在快速搜索框输入资源，点击搜索，则可查看资源的全部调用关系，包含资源与其他资源的调用关系、资源与广域网的调用关系以及资源自身的调用关系，对于搜索条件有更多的需求时，可通过设置高级搜索来实现

搜索条件可设置模块： - 资源集合：设置需要搜索的资源主体 - 资源外集合：查看资源集合与外部资源的调用关系 - 广域网集合：查看资源集合与广域网的调用关系 - 其他设置：作用于所有集合的公共搜索条件

分组条件

资源集合中的分组条件描述了统计数据的资源分组标准，由如下条件组成： - 资源分组：决定资源过滤匹配到的资源的分组标准。点击分组、拓扑节点、拓扑路径并进行资源分组的切换操作时，将会自动将该值设置为选中的资源类型并重新进行搜索。特别地，当资源分组选择 IP 地址时，允许按输入的掩码对 IP 地址进行聚合，默认不进行聚合。自动分组将自动识别资源，此资源为 IP 关联的资源 - 采集点分组：决定是否对不同的采集点分别统计 - 知识图谱：决定伴随资源展示的知识图谱信息，可选择资源池、计算资源、网络资源、网络服务、存储资源、容器资源、业务画像七个大会的十余种知识图谱属性 - 流特征分组：决定对资源分组的额外组合条件，包括 IP 类型、网络协议、服务端口 - 流日志特征分组：决定对资源分组的额外组合条件，包括流结束类型等 - HTTP 日志特征分组：决定对资源分组的额外组合条件，包括响应码等 - DNS 日志特征分组：决定对资源分组的额外组合条件，包括响应码等

资源过滤

决定如何对监控资源进行过滤，对所有页面均有效，支持从十余个维度对监控资源进行过滤。点击分组、拓扑节点、拓扑路径并进行资源分组的切换操作时，将会自动将该条件设置为选中的资源并重新进行搜索。

其他过滤

通过指定流量属性过滤条件，可以对匹配的流量做进一步过滤：

- 采集器：支持对流量经过的采集器进行过滤，可用于观测某个采集器上的所有监控数据，经常用于网关虚拟机位置
- 采集点：对流量的采集位置进行过滤，物理网络采集点可在资源-其他资源-采集点中进行维护
- IP 类型：对 IPv4、IPv6 流量进行过滤
- 监控资源角色：对监控资源的客户端、服务端角色进行过滤
- 网络协议：对流量的 L3 网络协议类型进行过滤
- 服务端口：对流量的 L4 服务端口进行过滤，支持 TCP 和 UDP 端口
- 路径数据：对端到端路径的类型进行过滤，支持指定广域网、资源内、资源外，分别表示资源集合与广域网间、资源集合内部、资源集合与资源外集合的流量
- 服务过滤：对服务端口进行过滤，当选择策略匹配服务时，仅展示容器服务相关的流量、资源组关键服务相关的流量以及负载均衡监听器相关的流量
- 指标量：通过选定一个指标量及其统计值范围，可以对统计结果中的分组做进一步的过滤
- 路径统计位置：表示流量采集、统计的位置，路径统计数据中支持如下位置（资源统计数据仅支持前两项）：
 - 客户端：流量主动发起方
 - 服务端：流量接收处理方
 - 客户端容器节点：客户端容器 POD 所在的容器节点
 - 服务端容器节点：服务端容器 POD 所在的容器节点
 - 客户端宿主机：客户端云服务器所在的宿主机
 - 服务端宿主机：服务端云服务器所在的宿主机
 - 客户端网关宿主机：客户端侧云服务器网关所在的宿主机，采集的流量更接近客户端则统计为客户端网关宿主机流量
 - 服务端网关宿主机：服务端侧云服务器网关所在的宿主机，采集的流量更接近服务端则统计为服务端网关宿主机流量
 - 客户端网关：客户端侧网关，采集的流量更接近客户端
 - 服务端网关：服务端侧网关，采集的流量更接近服务端
 - 网关：包含路由器、基于 MAC 路由的负载均衡器（例如微软云 MUX）
 - 其他：无法统计到以上罗列的位置中的都纳入其他，例如 Flood 到不相关宿主机上的流量
- 链路协议：可过滤流量的链路层协议（eth_type）属性
- 应用协议：可过滤流量的应用层协议属性，支持 HTTP、DNS
- 客户端口：可过滤传输层的客户端端口
- 流日志 ID：可根据 ID 过滤一条流的所有日志
- 客户端 MAC：可过滤流量的链路层的客户端 MAC 地址
- 服务端 MAC：可过滤流量的链路层的服务端 MAC 地址
- 隧道层数：可过滤存在隧道封装的流量
- 隧道类型：可过滤不同隧道封装协议的流量
- 请求隧道 ID：可过滤客户端发送给服务端流量的隧道 ID
- 响应隧道 ID：可过滤服务端发送给客户端流量的隧道 ID
- 请求隧道源 IP：可过滤客户端发送给服务端流量的隧道外层源 IP
- 请求隧道目的 IP：可过滤客户端发送给服务端流量的隧道外层目的 IP
- 响应隧道源 IP：可过滤服务端发送给客户端流量的隧道外层源 IP
- 响应隧道目的 IP：可过滤服务端发送给客户端流量的隧道外层目的 IP
- 流结束类型：可根据流结束类型过滤流量，例如：连接超时导致的结束、建连-客户端端口复用导致的结束等等
- 报文类型：根据 HTTP/DNS 报文的请求/响应/会话进行过滤，会话为合并同一条请求与

响应 - 协议版本：根据 HTTP 版本号过滤 - 请求方法：根据 HTTP 的请求方法过滤
- 代理客户端：根据 HTTP 的代理客户端（X-Forwarded-For）过滤 - 主机名：根据 HTTP 的主机名（host）过滤 - 路径：根据 HTTP 的路径（path）过滤 - 请求 ID：
根据 HTTP/DNS 的请求 ID 过滤 - TraceID：根据 HTTP 的 TraceID 过滤 - 响应码：
根据 HTTP/DNS 的响应码过滤 - 查询域名：根据 DNS 的查询域名过滤 - 查询类型：
根据 DNS 的查询类型过滤 - 解析地址：根据 DNS 的解析地址过滤

其他过滤支持正则匹配： - HTTP/DNS/SQL/NoSQL 等日志部分字段支持正则匹配：
=~、!~，即，模糊等于、模糊不等于，用来动态匹配数据 - 数字类型的部分字段支持 a-b/a-b,c 的搜索，例如服务端口、响应码等

模板变量

全景图支持为每一个历史搜索条件添加模板变量，模板变量可被搜索条件引用，通过模板变量切换来控制搜索条件

指标量

全景图支持 13 类 122 种指标量，与算子结合可产生 1867 个指标量统计值。除属性类指标量以外均支持 Avg（均值）、Max（峰值）、Min（谷值）、Percentile（百分位数）、Spread（绝对跨度）、RSpread（相对跨度）、StdDev（标准差）、Apdex（满意度）八种算子，除时延、负载和属性类指标量以外其他均支持 Sum（总量）算子，属性类指标量支持 Distinct（基数）算子。基于上述算子的计算结果可继续进行百分比和差量运算： - 吞吐 - 网络层吞吐 - 传输层吞吐 - 应用层吞吐 - 负载 - 传输层负载 - 时延 - 传输层时延 - 应用层时延 - 性能 - 传输层性能 - 异常 - 传输层异常 - 应用层异常 - 属性 - 资源属性 - 路径属性 - 其他属性 - 资源变更

详细指标量列表参考[全景图功能规格](#)。

子视图

流量搜索通过丰富多样的子视图，对流量数据进行可视化展现。每种子视图均支持一系列标准化操作，其中所有子视图均可支持的操作如下： * 添加到视图：将子视图添加到选定的视图中，视图中的名称默认复用在全景图中的名称 * 除了拓扑图以外，添加到视图中的所有子视图均支持生成报表 * 添加到视图中的所有折线图均支持设置告警 * 下载 CSV 数据：以 CSV 文件的形式下载子视图所使用的流量统计数据，用户可基于 CSV 文件进行二次开发 * 查看 API：查看获取子视图所使用的流量统计数据的 API，用户可通过调用 API 进行二次开发 * 全屏显示：全屏查看子视图，用于对可视区域进行放大，注意放大后子视图部分操作受限 * 弹框显示：弹框查看子视图，对子视图进行放大，且支持对子视图操作 * 切换数据源：除资源变更事件图以外，其他子视图均支持切换数据源，用户也可在系统-数据节点-全景图存储配置中修改和定义数据源 * 修改指标量：修改子视图默认显示的指标量，并设置阈值以便于快速判断指标量与警戒线的关系。不同类型的子视图对该项操作的支持程度存

在差别，详见下文介绍 * 样式设置：根据子视图的不同类型，支持对子视图的边距、图例显示、辅助线显示进行设置

部分子视图支持的操作有（指以资源/路径为维度展示的子视图）： * 点击某资源，支持的操作有：查看资源基本信息、查看资源关联信息、切换资源分组、查看调用关系、查看资源事件、查看指标曲线、查看日志详情、查看指标量 * 点击某路径，支持的操作有：查看资源基本信息、查看资源关联信息、切换资源分组、查看调用关系、查看资源事件、查看链路追踪、查看指标曲线、查看日志详情、查看指标量

异常提示： * 提示方式：子视图的左上方，示例如下 * 数据请求超时：当子视图数据请求超时时出现此提示，当再次刷新数据有响应时，该提示消失 * Traffic

Property Metric Error: Distribution not get available single metrics: 资源排名中的子视图选择路径属性指标量时出现此异常提示 * 查询数据节点失败：异常区域，数据节点名称：当查询数据节点异常时出现改提示

拓扑类

拓扑类子视图用于展示节点（资源）之间的通信关系（路径），以及资源和路径上的指标量大小。

基础拓扑图

未开启链路追踪时，流量拓扑展示为基础拓扑图。基础拓扑图由资源节点及有向路径组成，其资源节点分为两类： - 监控资源：由搜索条件中分组条件和资源过滤决定的节点 - 表征的资源类型：每个节点表示的资源类型（即图标类型）由搜索条件中的资源分组决定 - 图标外层的圆环：圆环大小表示主指标量统计值的大小，当圆表示的为资源外资源时显示为虚线，当统计值超过阈值时显示为红色 - 节点的悬停操作：展示资源的信息和指标量的统计值，以及分组条件中知识图谱框选择的额外资源属性信息 - 节点的点击操作：点击可查看资源基本信息、资源知识图谱、切换拓扑图展示的资源分组类型、切换至指标曲线页面 - **Internet:** 当监控资源与 **Internet** 资源存在通信时出现 - 表征的资源类型：表示 **DeepFlow** 资源-网络资源-子网/IP 页面展示的子网和 IP 之外的 IP 资源

基础拓扑图中的路径表示资源之间的客户端、服务端关系，路径的方向总是从客户端指向服务端： - 路径的颜色和粗细：路径的粗细表示主指标量统计值的大小，当统计值超过阈值时显示为红色 - 路径的悬停操作：展示两侧节点的关系（客户端->服务端），及在客户端/服务端、客户端/服务端容器节点、客户端/服务端宿主机、客户端/服务端网关宿主机、客户端/服务端网关侧分别采集到的指标量统计值和统计值最大最小的差量。统计位置未显示则可能因为不存在此位置也可能由于未部署采集器导致。统计值的差量可辅助判断路径上产生的网络丢包和时延。 - 路径的点击操作：与节点的点击弹出页相似

基础拓扑图支持如下标准化操作： - 基础操作 - 添加到视图 - 下载 CSV 数据 - 查看 API - 全屏显示 - 弹框显示 - 切换数据源 - 修改指标量：支持为所有指标量及其差量

设置阈值，超过阈值的节点外层圆环和路径将会显示为红色，对应的 tooltip 中会标红具体超出阈值的数据 - 其他操作 - 切换 Top：通过选择 Top 20 下拉框，可调整拓扑图中显示的监控资源点数量 - 最多展示 Top N 个客户端资源、Top N 个服务端资源 - 属于资源外集合中的资源将被绘制为虚线圈 - 展开表格：点击后展开所有资源和路径数据的统计表格，支持搜索、排序、过滤操作，点击表格的一行左侧拓扑图可同步高亮，再次点击表格一行可弹出资源信息显示页；点击左侧拓扑图中的资源或链路项，表格中相关资源与链路联动高亮 - 名称缺省显示/名称全称显示：点击后在全名和名称前缀之间切换，避免名称太长造成遮挡 - 开启链路追踪/关闭链路追踪：点击后切换基础拓扑图与全链路拓扑图 - 颜色设置：点击后可自定义基础拓扑图的圈/线颜色 - 补齐所有关联数据：默认状态下不做补齐，排名未进入 Top N 的数据无法展示，例如 Top N 的点可能缺少部分采集点的数据（选择按采集点分组时），Top N 的路径可能缺少部分统计位置或采集点的数据 - 显示资源逻辑关联：点击后根据负载均衡器规则、NAT 网关规则、资源组依赖关系自动补充负载均衡器与后端云服务器/IP、NAT 网关与 NAT 后的云服务器/IP、资源组之间的逻辑线，当存在流量线时则不补充逻辑线 - 显示特征关联关系：点击后补齐流/包特征节点与资源节点的连线 - 拓扑位置操作 - 放大缩小 - 保存：记忆节点位置，需要用户先保存搜索条件 - 随机：随机排布拓扑中的节点 - 自动：智能排布拓扑中的节点 - 图例：拓扑图图标说明，以及圆圈大小、线条粗细说明 - 手动补充资源关系模式：点击后进入手动补线模式，通过点击拓扑图上两个节点，则补充一条线，再次点击后回到拓扑图展示

全链路拓扑图

当开启链路追踪时，流量拓扑展示为全链路拓扑图。其下半部分为基础拓扑图，点击其中的节点和路径可在上半部分显示对应的虚拟和物理链路： - 物理链路： - 采集点分组=是，可查看物理链路，当基础拓扑图路径对应的流量的采集点为非虚拟网络时，则高亮对应的链路 - 路径的颜色和粗细：路径的粗细表示主指标量统计值的大小，当统计值超过阈值时显示为红色，否则显示为绿色 - 路径的悬停操作：展示两侧的物理网元，及在两个采集点分别采集到的指标量统计值。若两侧均有统计值，二者的偏差程度可辅助判断链路上的丢包和时延。 - 路径的点击操作：点击可查看物理网元的基本信息、切换至指标曲线页面查看与之关联的流量拓扑中资源间流量的历史统计值 - 点的颜色和粗细：点的粗细表示点连接的所有路径上主指标统计值的最大差量大小，可辅助判断网元上的丢包和时延。 - 点的悬停操作：展示网元的信息和相邻路径上的最大指标量差值 - 虚拟链路：基础拓扑图路径对应的虚拟网络流量统计位置 - 采集点分组=是，且存在虚拟网络采集点流量时，展示路径对应的虚拟链路 - 路径的悬停操作：展示两侧的统计位置的基础拓扑图路径指标量统计值。若两侧均有统计值，二者的偏差程度可辅助判断链路上的丢包和时延。 - 路径点击操作：弹出链路追踪详情图 - 点的悬停操作：展示当前统计位置的基础拓扑图路径指标量统计值 - 点点击操作：弹出链路追踪详情图

全链路拓扑图支持的标准操作相比基础拓扑图，加入： - 拓扑位置操作 - 分割线调整：物理拓扑图与基础拓扑图的分割线上下可调

知识图谱

点击资源或路径弹出框时，点击知识图谱，则将根据资源或路径的关联信息绘制星状的拓扑图 - 拓扑操作 - 点或者路径的悬浮：高亮关联的点与路径，并展示点的全名 - 点+按钮：将绘制当前点对应资源的知识图谱 - 其他操作：支持拓扑图放大缩小、查看图例、名称全显示、关联显示 - 拓扑逻辑： - 同类型的点，最多只展示 2 个，其他统一收入到\$num 个资源类型，比如 100 个虚拟机

链路追踪详情图

当点击全链路拓扑图的虚拟链路时，弹出链路追踪详情图，展示虚拟链路各个统计位置对应的资源、采集点、采集网卡标识、指标量数据等信息 - 链路追踪图上部分为虚拟链路，下部分为链路详情表/链路柱状图 - 链路柱状图：链路追踪详情图的差量数据无可视化能力，提供一种类柱状图的图形化展示能力 - 虚拟链路点击点或者路径时，可高亮链路详情表的对应的行 - 点击链路详情表的行，可出现弹出框，支持跳转到指标曲线和流量日志页面

链路追踪详情图支持的标准操作包括： - 基础操作 - 下载 CSV 数据 - 查看 API - 切换数据源 - 修改指标量：可选择指标量 - 切换显示指标量：按选中的指标量显示链路柱状图

非对称拓扑图

当高级搜索分组条件选择了流特征分组或包特征分组时，拓扑图将展示为非对称形式。具体来讲，图中每条路径有两种情况： - 资源 A -> 资源 B+特征分组信息：A 作为客户端访问 B，并携带该特征分组信息 - 资源 B+特征分组信息 -> 资源 B：为了将资源 B 的各种特征分组信息节点汇集起来显示的虚拟路径，无统计数据。但是若资源 B 在整个拓扑中只有一个特征分组信息，则会自动隐藏这条连线（即将上述两个资源节点合并为一个节点）。

特别地，当其他分组条件选择服务端口时，拓扑图中可展示端口号对应的容器服务名、资源组关键服务名、端口号知名服务名。

非对称拓扑图支持的标准操作与基础拓扑图相同。

比较类

比较类子视图用于展示不同资源（分组）的指标量统计值排名、占比等信息。

柱状图

可显示 Top N 分组信息（资源、端到端访问关系、资源/端到端访问关系+其他分组条件等）。每个分组均支持鼠标悬停和点击，且其效果与拓扑图中的悬停和点击类似。

柱状图支持的标准操作包括： - 基础操作 - 添加到视图 - 下载 CSV 数据 - 查看 API - 全屏显示 - 弹框显示 - 切换数据源 - 修改指标量：支持为主指标量设置阈值，设置阈值时图中会显示一条阈值虚线，超出阈值的数据点 tooltip 中会标红显示 - 其他操作 - 切换 Top：通过选择 Top 5 下拉框，可调整显示的分组数量 - 名称缺省显示/名称全称显示：切换是否展示 Top 资源或路径的全名 - 切换图表：可切换显示为竖直柱图或水平柱图 - 切换展示方式：可切换至饼状图、Top 折线图 - 升序排序/降序排序：默认为降序排列，可实现柱图从大到小、从小到大排列

饼状图

饼状图与柱状图的含义、出现的页面均非常相似，他们之间的主要差异为：饼状图可显示 Top N 分组的统计值占比，并支持显示其他分组。

饼状图支持的标准操作包括： - 基础操作 - 添加到视图 - 下载 CSV 数据 - 查看 API - 全屏显示 - 弹框显示 - 切换数据源 - 修改指标量：可选择指标量，不支持设置阈值 - 其他操作 - 切换 Top：通过选择 Top 5 下拉框，可调整显示的分组数量 - 名称缺省显示/名称全称显示：切换是否展示资源或路径的全名 - 切换图表：可切换显示为实心饼状图或空心环状图 - 切换展示方式：可切换至柱状图、Top 折线图

Top 折线图

Top 折线图与柱状图、饼状图出现的页面相同，他们的主要差异为：Top 折线图可显示分组的统计值历史曲线。

Top 折线图支持的标准操作包括： - 基础操作 - 添加到视图 - 下载 CSV 数据 - 查看 API - 全屏显示 - 弹框显示 - 切换数据源 - 修改指标量：支持为主指标量设置阈值，设置阈值时图中会显示一条阈值虚线，超出阈值的数据点 tooltip 中会标红显示 - 设置滑窗：支持通过滑窗对告警事件进行抑制，避免抖动的指标量统计值频繁触发告警 - 设置滑窗后：某个时间点 t 的数值为 $[t - interval * window\_size + interval, t + interval)$ 这段时间内所有数据点在某算子下的聚合值；特别的，当 `window_size` 为 1（默认值）时，折线图为原图 - 其他操作 - 名称缺省显示/名称全称显示：切换是否展示资源或路径的全名 - 切换图表：可切换显示为折线或梯形线 - 切换 Top：通过选择 Top 5 下拉框，可调整显示的分组数量 - 切换展示方式：可切换至柱状图、饼状图 - 切换补值方式：可将数据点为空的值填入 0、null（时间点无值）、none（无时间点） - 序列选择：可选择过滤需要展示的序列数据 - 展示图例表格/隐藏图例表格：切换是否展示表格形式的图例及其摘要统计值 - 展示所有指标量/展示主指标量：切换是否将所有指标量展示为单独的序列 - 切换堆叠/切换平铺：切换是否采用堆叠的方式展示多个序列 - 选择时间范围：鼠标框选一个时间范围，整个页面将会刷新 - 升序排序/降序排序：默认为降序排列，可实现柱图从大到小、从小到大排列（升序排列时无统计值的资源无法触发告警）

表格

表格一般作为柱状图、饼状图的详情展示。表格的每一行均支持点击，点击效果与拓扑图中的点击一致。

表格支持的标准操作包括： - 基础操作 - 添加到视图 - 下载 CSV 数据 - 查看 API - 全屏显示 - 弹框显示 - 切换数据源 - 修改指标量：支持为所有指标量设置阈值，超过阈值的统计值将会在表格中显示为红色 - 其他操作 - 调整分页：通过选择每页数、页码调整表格的分页显示 - 选择显示列：支持选择展示的列 - 列宽重置：平均分配每列列宽

分布类

概率分布图

概率分布图用于显示 Top N 分组的指标量统计值的概率分布，显示的曲线可近似描述统计值的概率密度函数（PDF，Probability Density Function），其 X 轴表示统计值区间，Y 轴表示统计值落在该区间的分组占有所有分组的百分比。

概率分布图支持的标准操作包括： - 基础操作 - 添加到视图 - 全屏显示 - 弹框显示 - 切换数据源 - 下载 CSV 数据：下载的数据为资源分组的指标量统计值，不是统计值的概率分布 - 查看 API：调用显示的 API，获取到的未 Top N 分组的指标量统计值，不是统计值的概率分布 - 修改指标量：仅支持选择一个指标量，不支持设置阈值 - 其他操作 - 名称缺省显示/名称全称显示：切换是否展示资源或路径的全名 - 切换 Top：通过选择 Top 100 下拉框，可调整用于计算概率分布的分组数量 - 支持框选若干柱，进行取值范围二次过滤查询

趋势类

趋势类子视图用于展示特定资源的指标量统计值随时间的变化。

趋势分析图

趋势分析图展示监控资源指标量随时间的变化趋势，辅助用户缩小时间范围。

趋势分析图支持的标准操作包括： - 基础操作 - 添加到视图 - 切换数据源 - 下载 CSV 数据 - 查看 API - 全屏显示 - 弹框显示 - 修改指标量：支持为主指标量设置阈值，超过阈值的柱展示为红色，对应的 tooltip 显示标红 - 其他操作 - 名称缺省显示/名称全称显示：切换是否展示资源全名 - 序列选择：可选择过滤需要展示的序列数据 - 切换补值方式：可将数据点为空的值填入 0、null（时间点无值）、none（无时间点） - 切换堆叠/切换平铺：切换是否采用堆叠的方式展示多个序列 - 展示所有指标量/展示主指标量：切换是否将所有指标量展示为单独的序列 - 选择时间范围：鼠标框选一个时间范围，整个页面将会刷新

资源变更事件图

用于展示监控资源集合的变更事件，包括资源创建（同步）、启停、删除、迁移、IP 变更等事件。资源动态性是云环境的主要特定，且在容器环境下表现更为突出。通过展示资源变更事件统计数据 and 事件详情，用户可以与页面上的折线图进行关联分析，快速定位网络问题。

资源变更事件图由柱状图和表格组成，柱状图支持的标准操作包括：- 基础操作 - 添加到视图 - 下载 CSV 数据 - 查看 API - 全屏显示 - 弹框显示 - 其他操作 - 切换图表：可切换为显示堆叠柱或并行柱 - 序列选择：可选择过滤需要展示的序列数据 - 切换堆叠/切换平铺：切换是否采用堆叠的方式展示多个序列 - 查看列表：展示表格弹出页 - 选择时间范围：鼠标框选一个时间范围，整个页面将会刷新

资源类型及记录的变更事件关系如下：

资源类型	变更事件
云服务器	创建/同步、删除、增加 IP、删除 IP、状态改变、迁移
路由器	创建/同步、删除、增加 IP、删除 IP
DHCP 网关	创建/同步、删除、增加 IP、删除 IP
容器 POD	创建/同步、删除、增加 IP、删除 IP、状态改变、迁移、重建
容器服务	创建/同步、删除、增加 IP、删除 IP
容器节点	创建/同步、删除、增加 IP、删除 IP
RDS	创建/同步、删除、增加 IP、删除 IP
Redis	创建/同步、删除、增加 IP、删除 IP
负载均衡器	创建/同步、删除、增加 IP、删除 IP
NAT 网关	创建/同步、删除、增加 IP、删除 IP

折线图

折线图中可展示同一个资源在不同采集点的统计值。

折线图支持的标准操作包括：- 基础操作 - 添加到视图 - 下载 CSV 数据 - 查看 API - 全屏显示 - 弹框显示 - 切换数据源 - 修改指标量：支持为主指标量设置上下限阈值，设置阈值时图中会显示对应的阈值虚线，超出上限或者低于下限的数据点，**tooltip** 上会以标红形式显示 - 设置滑窗：支持通过滑窗对告警事件进行抑制，避免抖动的指标量统计值频繁触发告警 - 其他操作 - 名称缺省显示/名称全称显示：切换是否展示资源全名 - 切换图表：可切换显示为折线或梯形线 - 序列选择：可选择过滤需要展示的序列数据 - 切换补值方式：可将数据点为空的值填入 0、null（时间点无值）、none（无时间点） - 展示图例表格/隐藏图例表格：切换是否展示表格形式的图例及其摘要统计值 - 展示所有指标量/展示主指标量：切换是否将所有指标量

展示为单独的序列 - 切换堆叠/切换平铺：切换是否采用堆叠的方式展示多个序列 - 选择时间范围：鼠标框选一个时间范围，整个页面将会刷新

流量日志

从指标数据可跳转至日志详情页面，查看对应的流量日志详情，提供回溯取证的能力，流量日志详情也支持跳转到聚合页面，可对日志数据聚合分析

异常提示： - 某区域内的数据查询不全时，流量日志页面子视图趋势图、表格出现异常提示查询数据异常 - 当整个区域内的数据查询都失败时，区域 tab 页右侧提示用户查询数据异常异常区域：\$区域名称

流日志

流日志数据是基于五元组聚合存储的流数据，对于五元组对应的连接超过 1 分钟的情况会每隔 1 分钟输出一次当前流的信息。

相比指标数据，流日志能提供的额外能力为： - 查看每条流的详细属性和指标量统计值，总共支持查看近 100 个参数 - 查看每条流的源端口号 - 查看每条流的双端 MAC 地址、链路协议、VLAN - 查看每条流的隧道类型、隧道 ID、双端隧道 IP - 查看每条流的双端真实 IP，即使是 Internet 上的 IP（在指标数据中记录为 0.0.0.0），流日志中也会进行记录 - 查看每条流的应用协议，包括 HTTP、DNS 和其他

HTTP 日志

HTTP 日志是基于客户端与服务端之间的 HTTP 流量提供的日志，可查看 HTTP 流量中报文类型、协议版本、请求方法、代理客户端、响应码以及请求与响应包时延等信息

DNS 日志

DNS 日志是基于客户端与服务端之间的 DNS 流量提供的日志，可查看 DNS 流量中报文类型、查询域名、解析地址、响应码以及请求与响应包时延等信息

典型场景

业务性能

钻取排查业务慢的根因

业务性能问题：访问服务（IP 地址 10.20.30.40）慢。

问题分析：故障信息有限，需要避免不同 VPC 内相同内网 IP 的干扰，并定位故障可能的具体位置（虚拟网络客户端侧、虚拟网络服务端侧、物理网络具体网元、运营商异常）和历史时间点。

排查步骤：

- 在指标数据中进行搜索
 - 填写搜索条件
 - 分组条件：资源分组=网络/IP 地址
 - 资源过滤：IP 地址=10.20.30.0/24
 - 查看流量拓扑，选择传输层时延或传输层性能的指标量
 - 点击 IP 10.20.30.40 节点，查看关联的子网、云服务器信息：子网 A、云服务器 A
 - 增加资源过滤条件：子网=子网 A、云服务器=云服务器 A
- 若不了解具体服务端口
 - 修改搜索条件，流特征分组选中服务端口
 - 一般而言吞吐量最大的服务是主要服务，假设为 80
 - 修改搜索条件，增加流量属性过滤条件服务端口=80
- 在指标数据页面定位虚拟网络的异常
 - 选择拓扑图的指标量为传输层时延或传输层性能，查看时延、重传、零窗是否异常，定位异常的路径
 - 查看路径上的指标量统计值，对比客户端和服务端的统计值差异，判断哪一侧存在异常
- 在物理拓扑部分定位物理网络的异常
 - 若客户端和服务端的统计数据均没有异常，点击异常的路径展示物理拓扑上的指标量
 - 逐跳查看物理网络上不同采集点的性能统计数据，确定异常点
- 查看指标曲线定位故障的具体时间
 - 上述步骤中，均可通过点击弹出框查看指标曲线
 - 确认指标量统计值的异常时间是否和故障时间匹配
 - 调整时间范围，确定其他时间是否有故障
 - 调整数据源为秒级，确定是否为微突发或短时抖动

完成故障定位之后一般需要留存证据，此时可跳转至其他页面进行取证：

- 打开全景图-网络拓扑页面，从网络配置、连通性上进行取证
- 跳转查看日志详情页面，从流量日志上进行取证
- 打开全景图-流量下载页面，下载故障期间的 PCAP 包（需要事先创建 PCAP 下载策略）

若本次故障未找到根因，可通过下述措施为下次定位做准备：

- 将故障排查中的关键曲线图添加到视图中，并在视图中设置关键指标量的告警
- 在全景图-流量下载页面配置合适的 PCAP 下载策略

定位访问最慢的客户端云服务器

业务性能问题：找出访问服务（云服务器 A、服务端口 443）最慢的 10%客户端云服务器。

问题分析：直接对时延进行排序是最简单的操作，但同时需要排除小流量资源对统计结果的干扰

排查步骤：

- 在指标数据页面中进行搜索
 - 填写搜索条件
 - 分组条件：资源分组=云服务器
 - 资源过滤：VPC=云服务器 A 所在的 VPC
 - 流量属性过滤条件：服务端口=443
 - 查看路径流量详情表格，同时选择传输层时延、吞吐量指标量
 - 按 TCP 时延逆序排列，查看最慢的 10%云服务器流量与其他云服务器流量差异是否明显
 - 若最慢的 10%客户端云服务器存在流量很小的情况
 - 增加指标量过滤条件：选择合适的吞吐量统计值、填写合适的阈值过滤流量过小的客户端

定位容器服务后端的异常 POD

业务性能问题：找出容器服务（Service A）后端是否有吞吐量明显小的 POD

问题分析：容器服务通常以等概率的方式向后端 POD 分发业务请求，正常情况下所有后端 POD 的吞吐量应该一致。为了定位容器服务的后端 POD，首先需要到业务画像中选择该容器服务创建对应的后端 POD 资源组（Group A）。

排查步骤：

- 在指标数据页面中进行搜索
 - 填写搜索条件
 - 分组条件：资源分组=POD
 - 资源过滤：VPC=容器集群对应的 VPC
 - 查看路径流量详情表格，以吞吐量逆序排序，确认是否有 POD 的吞吐量明显低
 - 修改分组条件：资源分组=容器节点，确认是否有容器节点的吞吐量明显低
 - 点击容器节点，确认节点上的容器 POD 数量是否明显低
 - 若节点上容器 POD 数量与其他节点差异不大，需要排查是否是容器节点存在问题

- 点击异常 POD 或异常节点，跳转查看指标曲线页面
 - 查看网络层吞吐和传输层时延/性能/异常，排查吞吐量低和其他指标量是否存在时间上的关联
- 查看资源盘点
 - 切换监控资源分组为容器节点或容器 POD
 - 确认是否有未分发流量的后端 POD 或容器节点
 - 未分发流量可能由于 POD 状态不正常导致

资源用量

虚拟交换机的异常 Flood

资源用量问题：找出哪个子网的 BUM（广播、组播、未知单播）最大

问题分析：BUM 可能导致虚拟交换机进行大量复制，正常情况下网络中的 BUM 速率应该很低

操作步骤：

- 在指标数据页面进行搜索
 - 填写搜索条件
 - 分组条件：资源分组=子网
 - 其他过滤：统计位置=其他
 - 查看表格，选择指标量为字节均值、包数均值、字节峰值、包数峰值
 - 按均值逆序排列，查看是否有子网的统计值异常高
 - 按峰值逆序排列，与均值对比判断是否有子网的统计值异常高
- 跳转查看指标曲线页面
 - 切换数据源为秒级，确认 Flood 是否有时间特征
- 返回指标数据页面
 - 切换资源分组为宿主机、云服务器、容器节点、容器 POD，发现导致 Flood 的具体资源

盘点闲置的计算资源

资源用量问题：找出闲置的计算资源（云服务器）

问题分析：闲置的计算资源 CPU、内存消耗可能没有特征，但流量一般存在特征。一个活跃的资源在相当长的时间范围内肯定会产生流量，且一般都会向其他资源提供服务。另外，资源用量一般需要生成周期性报表，并可能会对数据进行二次开发。

操作步骤：

- 在指标数据页面进行搜索
 - 填写搜索条件

- 分组条件：资源分组=云服务器
- 时间控件范围选择最近一个月，查看表格并按字节总量增序排列，确认是否有流量为 0 的计算资源
- 增加搜索条件的流量属性过滤条件：监控资源角色=服务端
- 查看表格确认是否有不提供任何服务的计算资源
- 点击资源盘点表格的下载 CSV 数据，可将所有数据下载后进行二次处理
- 点击资源盘点表格的添加到视图，可在视图中配置周期性的报表生成策略，避免查询一个月的数据等待时间太长

统计广域网带宽的消耗

资源用量问题：找出消耗广域网带宽最大的 VPC

问题分析：带宽消耗一般需要考虑上行、下行方向

操作步骤：

- 在指标数据页面进行搜索
 - 填写搜索条件
 - 分组条件：资源分组=VPC
 - 其他过滤：监控资源角色=客户端，路径=广域网
 - 查看路径流量详情表格，按接收字节总量逆序排列，找出作为客户端请求广域网消耗下行带宽的 VPC 排名
 - 修改搜索条件
 - 流量属性过滤条件：监控资源角色=服务端
 - 查看路径流量详情表格，按发送字节总量逆序排列，找出作为服务端响应广域网消耗上行带宽的 VPC 排名

统计广域网 IPv6 带宽的消耗

资源用量问题：找出消耗广域网 IPv6 带宽最大的 VPC

问题分析：带宽消耗一般需要考虑上行、下行方向

操作步骤：

- 在指标数据页面进行搜索
 - 填写搜索条件
 - 分组条件：资源分组=VPC
 - 其他过滤：IP 类型=IPv6，监控资源角色=客户端，路径=广域网
 - 查看路径流量详情表格，按接收字节总量逆序排列，找出作为客户端请求广域网消耗下行带宽的 VPC 排名
 - 修改搜索条件

- 流量属性过滤条件：监控资源角色=服务端
- 查看路径流量详情表格，按发送字节总量逆序排列，找出作为服务端响应广域网消耗上行带宽的 VPC 排名

统计专线带宽的消耗

资源用量问题：找出消耗专线带宽最大的 VPC

问题分析：带宽消耗一般需要考虑上行、下行方向。在搜索之前，需要将专线对端的子网录入至资源管理-网络资源中，记为子网 A。

操作步骤：

- 在指标数据页面进行搜索
 - 填写搜索条件
 - 分组条件：资源分组=子网
 - 资源过滤：子网=子网 A
 - 其他过滤：监控资源角色=服务端，路径=资源集内
 - 查看路径流量详情表格，按发送字节总量逆序排列，找出作为客户端请求专线子网 A 消耗下行带宽的 VPC 排名
 - 修改搜索条件
 - 流量属性过滤条件：监控资源角色=客户端
 - 查看路径流量详情表格，按接收字节总量逆序排列，找出作为服务端响应专线子网 A 消耗上行带宽的 VPC 排名

安全隐患

发现 DoS/Flood 攻击

安全隐患问题：网络中是否有 TCP SYN Flood 正在发生

问题分析：SYN Flood 容易导致半开或重置连接数增大

操作步骤：- 在指标数据页面进行搜索 - 填写搜索条件 - 分组条件：资源分组=IP - 查看路径流量详情表格，按传输层异常中的半开或重置逆序排列，找出连接数最大的客户端 A - 若源端角色为客户端，此时源端为客户端 - 若源端角色为服务端，此时目的端为客户端

发现慢速端口扫描

安全隐患问题：网络中是否有端口扫描正在发生

问题分析：端口扫描在流量上的体现为恶意服务端请求了很多服务端口

操作步骤：- 在指标数据页面进行搜索 - 选择时间范围最近 7 天 - 填写搜索条件 - 分组条件：资源分组=IP - 查看路径流量详情表格，按其他属性中的服务端口排列，找

出可以客户端 - 若源端角色为客户端，此时源端为客户端 - 若源端角色为服务端，此时目的端为客户端 - 查看资源流量详情表格，按传输层异常中的半开、重置逆序排列，确认客户端 A 是否产生了大量的半开或重置连接

发现控制台暴力破解尝试

安全隐患问题：网络中是否有控制台爆破尝试正在发生

问题分析：控制台爆破针对 22（SSH）、23（Telnet）、3389（RDP）等端口。

操作步骤： - 在指标数据页面进行搜索 - 填写搜索条件 - 分组条件：资源分组=IP - 其他过滤：服务端口=22,23,3389 - 查看路径流量详情表格，按新建连接逆序排列，找出连接数最大的客户端 A - 若源端角色为客户端，此时源端为客户端 - 若源端角色为服务端，此时目的端为客户端

过滤记录流日志的采集点

目前一个数据节点写入流日志的性能默认为 1K/s，超过这个吞吐量时将会对数据进行采样写入。为了避免采样，若用户仅需要对特定采集点（例如互联网出口）的流量记录流日志，可通过配置文件配置采集点过滤条件。

在系统-采集器-配置中修改全景图功能开关-过滤流日志可对记录流日志的采集点进行过滤。

功能规格

规格

项目	规格	修改限制
分钟级数据保留时长	7 天	系统管理-数据节点-全景图存储配置页面
秒级数据保留时长	1 天	系统管理-数据节点-全景图存储配置页面
一个子视图内可展示的统计值种数	10 个	无法修改
选择框多选上限数目	25 个	前端配置文件 /var/www/lcweb/public/app_config_user.js 中的 FLOW_VIEW_MULTIPLE_SELECT_LEN
每个数据节点各类流量日志的写入速率	50k/s	见流量日志写入限速章节

授权

全景图功能需要获取 DeepFlow 监控诊断授权，请向云杉网络服务人员申请。

限制

- 数据源
 - 分钟级数据源在计算速率类的指标量时，峰值和谷值是通过每分钟的数据除以 60 秒得到的
- NetFlow/NetStream/sFlow
 - 对吞吐量中字节、包数的统计会乘以采样比，但是部分交换机可能不支持在 NetStream/NetFlow v5 包中携带 Sampling Interval 字段
 - 对连接数的统计不会乘以采样比，一个 sFlow/NetFlow 包中的原始包/流，仅记作一个连接
 - 采集器以收到 sFlow 包时的本地时间作为其中包的时间，以避免交换机与采集器时间不同步
 - 采集器以收到 NetFlow/NetStream 包时的本地时间作为其中 Flow 的结束时间，并以该时间作为流量数据的统计时间，以避免交换机与采集器时间不同步或发送时间滞后
- 流量采集
 - 对于 VXLAN 封装的业务流量，为了和分发的 VXLAN 封装流量区分以避免环路，采集器将会过滤掉 VXLAN 包头中第一个字节为 0xff 的流量
- 流统计-资源盘点
 - 资源盘点不会统计无 IP 地址的资源，例如没有 IP 地址的云服务器、空 VPC、空子网，此类空资源可在资源管理页面中通过排序的方法直接得到
- 指标量精度
 - 对于有除法运算的指标量如平均包长、异常比例、相对跨度（RSpread）算子等，页面统一保留两位有效数字显示，因此小于 0.01 的误差在页面上无法体现
- 指标量中发送、接收的方向说明
 - 涉及到路径（A 到 B）的场景下，发送、接收均以客户端（即 A）为视角
 - 其他情况下，发送、接收均以选定的资源为视角
- 对 HTTP 1.x 和 HTTP 2 进行解析的限制
 - 采集器上仅会分析 HTTP 头所在的包，HTTP 跨越了两个包时无法解析
 - 对 HTTP 2 的时延计算中，若同一个五元组中并发请求数量大，只能同时计算 8 个请求的时延
- 对 DNS 进行解析的限制
 - 若同一个五元组中并发请求数量大，只能同时计算 8 个请求的时延

- 对属性类指标量的限制
 - 查询资源的指标量时，不能选择路径属性类指标量
 - 查询路径的指标量时，不能选择资源属性类指标量
- 当资源分组选择 IP 地址且指定了聚合地址的掩码时，不支持查看关联的资源组及资源组服务信息

建议： - 建议用户配置 NetFlow/NetStream 的采样间隔为 0，以避免部分交换机不支持传输 Sampling Interval 时导致页面显示的流量统计值仅为采样后的值 - 建议用户配置 NetFlow/NetStream 的活跃流老化时间为 1 分钟，以避免造成采集器统计数据的精度过低 - 建议用户配置 NetFlow/NetStream 的流模板刷新闻隔为 1 分钟，以避免造成采集器重启时长时间无法确定流模板

具体交换机配置可参考部署扩容-采集器部署-专属服务器。

规范

算子列表

- Avg: 一段时间内该指标量所有取值的平均值
- Sum: 一段时间内该指标量所有取值的总和
- Max: 一段时间内该指标量所有取值的最大值
- Min: 一段时间内该指标量所有取值的最小值
- Percentile: 一段时间内该指标量所有取值的百分位数，例如 Percentile 90 表示一段时间内该指标量取值从小到大排列后位于 90%位置的数值
- Spread: 一段时间内该指标量所有取值的绝对跨度，即这些取值中的 Max-Min
- RSpread: 一段时间内该指标量所有取值的相对跨度，即这些取值中的 Max/Min
- StdDev: 一段时间内该指标量所有取值的标准差

指标量列表

注：Percentile（简写为 PCTL）的统计值名称后缀是具体的百分位参数值，下述文档以 95 为例。

吞吐

吞吐量从字节、包、连接等角度统计网络中传输的数据量。

网络层吞吐

指标量	请求字段	算子	响应字段	统计值	单位
字节	byte	Sum	sum_byte	字节总量	字节
		Avg	avg_bps	比特均值	比特/秒

发送字节	byte_tx	Max	max_bps	比特峰值	比特/秒
		Min	min_bps	比特谷值	比特/秒
		PCTL	percentile_bps	比特 P95	比特/秒
		Spread	spread_bps	比特绝对跨度	比特/秒
		RSpread	rsread_bps	比特相对跨度	
		StdDev	stddev_byte	字节标准差	字节
		Sum	sum_byte_tx	发送字节总量	字节
		Avg	avg_bps_tx	发送比特均值	比特/秒
		Max	max_bps_tx	发送比特峰值	比特/秒
		Min	min_bps_tx	发送比特谷值	比特/秒
接收字节	byte_rx	PCTL	percentile_bps_tx	发送比特 P95	比特/秒
		Spread	spread_bps_tx	发送比特绝对跨度	比特/秒
		RSpread	rsread_bps_tx	发送比特相对跨度	
		StdDev	stddev_byte_tx	发送字节标准差	字节
		Sum	sum_byte_rx	接收字节总量	字节
		Avg	avg_bps_rx	接收比特均值	比特/秒
		Max	max_bps_rx	接收比特峰值	比特/秒
		Min	min_bps_rx	接收比特谷值	比特/秒
		PCTL	percentile_bps_rx	接收比特 P95	比特/秒
		Spread	spread_bps_rx	接收比特绝对跨度	比特/秒

包数	packet	RSpread	rspread_bps_rx	接收比特相对跨 度	
		StdDev	stddev_byte_rx	接收字节标准差	字节
		Sum	sum_packet	包数总量	包
		Avg	avg_pps	包数均值	包/ 秒
		Max	max_pps	包数峰值	包/ 秒
		Min	min_pps	包数谷值	包/ 秒
		PCTL	percentile_pps	包数 P95	包/ 秒
		Spread	spread_pps	包数绝对跨度	包/ 秒
		RSpread	rspread_pps	包数相对跨度	
		StdDev	stddev_packet	包数标准差	包
发送包数	packet_tx	Sum	sum_packet_tx	发送包数总量	包
		Avg	avg_pps_tx	发送包数均值	包/ 秒
		Max	max_pps_tx	发送包数峰值	包/ 秒
		Min	min_pps_tx	发送包数谷值	包/ 秒
		PCTL	percentile_pps_tx	发送包数 P95	包/ 秒
		Spread	spread_pps_tx	发送包数绝对跨 度	包/ 秒
		RSpread	rspread_pps_tx	发送包数相对跨 度	
		StdDev	stddev_packet_tx	发送包数标准差	包
		Sum	sum_packet_rx	接收包数总量	包
		Avg	avg_pps_rx	接收包数均值	包/ 秒
接收包数	packet_rx	Max	max_pps_rx	接收包数峰值	包/ 秒

网络层载 荷	l3_byte	Min	min_pps_rx	接收包数谷值	包/ 秒
		PCTL	percentile_pps_rx	接收包数 P95	包/ 秒
		Spread	spread_pps_rx	接收包数绝对跨 度	包/ 秒
		RSpread	rsread_pps_rx	接收包数相对跨 度	
		StdDev	stddev_packet_rx	接收包数标准差	包
		Sum	sum_l3_byte	网络层载荷总量	字节
		Avg	avg_l3_bps	网络层载荷均值	比特 /秒
		Max	max_l3_bps	网络层载荷峰值	比特 /秒
		Min	min_l3_bps	网络层载荷谷值	比特 /秒
		PCTL	percentile_l3_bps	网络层载荷 P95	比特 /秒
发送网络 层载荷	l3_byte_tx	Spread	spread_l3_bps	网络层载荷绝对 跨度	比特 /秒
		RSpread	rsread_l3_bps	网络层载荷相对 跨度	
		StdDev	stddev_l3_byte	网络层载荷标准 差	字节
		Sum	sum_l3_byte_tx	发送网络层载荷 总量	字节
		Avg	avg_l3_bps_tx	发送网络层载荷 均值	比特 /秒
		Max	max_l3_bps_tx	发送网络层载荷 峰值	比特 /秒
		Min	min_l3_bps_tx	发送网络层载荷 谷值	比特 /秒
		PCTL	percentile_l3_bps_tx	发送网络层载荷 P95	比特 /秒
		Spread	spread_l3_bps_tx	发送网络层载荷	比特

接收网络 层载荷	l3_byte_rx	RSpread	rspread_l3_bps_tx	绝对跨度 发送网络层载荷 相对跨度	/秒
		StdDev	stddev_l3_byte_tx	发送网络层载荷 标准差	字节
		Sum	sum_l3_byte_rx	接收网络层载荷 总量	字节
		Avg	avg_l3_bps_rx	接收网络层载荷 均值	比特 /秒
		Max	max_l3_bps_rx	接收网络层载荷 峰值	比特 /秒
		Min	min_l3_bps_rx	接收网络层载荷 谷值	比特 /秒
		PCTL	percentile_l3_bps_rx	接收网络层载荷 P95	比特 /秒
		Spread	spread_l3_bps_rx	接收网络层载荷 绝对跨度	比特 /秒
		RSpread	rspread_l3_bps_rx	接收网络层载荷 相对跨度	
		StdDev	stddev_l3_byte_rx	接收网络层载荷 标准差	字节
平均包长	bpp	Sum	sum_avg_bbp	累计平均包长	字节
		Avg	avg_avg_bbp	平均包长均值	字节
		Max	max_avg_bbp	平均包长峰值	字节
		Min	min_avg_bbp	平均包长谷值	字节
		PCTL	percentile_div_bbp	平均包长 P95	字节
		Spread	spread_bbp	平均包长绝对跨 度	字节
		RSpread	rspread_bbp	平均包长相对跨 度	
		StdDev	stddev_avg_bbp	平均包长标准差	字节
平均发送 包长	bpp_tx	Sum	sum_avg_bbp_tx	累计平均发送包 长	字节
		Avg	avg_avg_bbp_tx	平均发送包长均 值	字节

平均接收包长	bpp_rx	Max	max_avg_bbp_tx	平均发送包长峰值	字节
		Min	min_avg_bbp_tx	平均发送包长谷值	字节
		PCTL	percentile_avg_bbp_tx	平均发送包长P95	字节
		Spread	spread_bbp_tx	平均发送包长绝对跨度	字节
		RSpread	rsread_bbp_tx	平均发送包长相对跨度	
		StdDev	stddev_avg_bbp_tx	平均发送包长标准差	字节
		Sum	sum_avg_bbp_rx	累计平均接收包长	字节
		Avg	avg_avg_bbp_rx	平均接收包长均值	字节
		Max	max_avg_bbp_rx	平均接收包长峰值	字节
		Min	min_avg_bbp_rx	平均接收包长谷值	字节
		PCTL	percentile_avg_bbp_rx	平均接收包长P95	字节
		Spread	spread_bbp_rx	平均接收包长绝对跨度	字节
		RSpread	rsread_bbp_rx	平均接收包长相对跨度	
		StdDev	stddev_avg_bbp_rx	平均接收包长标准差	字节

传输层吞吐

指标量	请求字段	算子	响应字段	统计值	单位
新建连接	new_flow	Sum	sum_new_flow_count	新建连接总量	连接
		Avg	avg_new_fps	新建连接均值	连接/秒
		Max	max_new_fps	新建连接峰值	连接/秒

关闭连接	closed_flow	Min	min_new_fps	新建连接谷值	连接/秒
		PCTL	percentile_new_fps	新建连接 P95	连接/秒
		Spread	spread_new_fps	新建连接相对跨度	连接/秒
		RSpread	rsread_new_fps	新建连接绝对跨度	
		StdDev	stddev_new_flow_count	新建连接标准差	连接
		Sum	sum_closed_flow_count	关闭连接总量	连接
		Avg	avg_closed_fps	关闭连接均值	连接/秒
		Max	max_closed_fps	关闭连接峰值	连接/秒
		Min	min_closed_fps	关闭连接谷值	连接/秒
		PCTL	percentile_closed_fps	关闭连接 P95	连接/秒
		Spread	spread_closed_fps	关闭连接相对跨度	连接/秒
		RSpread	rsread_closed_fps	关闭连接绝对跨度	
		StdDev	stddev_closed_flow_count	关闭连接标准差	连接

应用层吞吐

指标量	请求字段	算子	响应字段	统计值	单位
HTTP 请求	http_request	Sum	sum_packet_http_request	HTTP 请求总量	个
		Avg	avg_pps_http_request	HTTP 请求均值	个/秒
		Max	max_pps_http_request	HTTP 请求峰值	个/秒

HTTP 请求	http_request	Min	min_pps_http_request	HTTP 请求谷值	个 / 秒
		PCTL	percentile_pps_http_request	HTTP 请求 P95	个 / 秒
		Spread	spread_pps_http_request	HTTP 请求绝对跨度	个 / 秒
		RSpread	rspread_pps_http_request	HTTP 请求相对跨度	
		StdDev	stddev_packet_http_request	HTTP 请求标准差	个
	http_response	Sum	sum_packet_http_response	HTTP 响应总量	个
		Avg	avg_pps_http_response	HTTP 响应均值	个 / 秒
		Max	max_pps_http_response	HTTP 响应峰值	个 / 秒
		Min	min_pps_http_response	HTTP 响应谷值	个 / 秒
		PCTL	percentile_pps_http_response	HTTP 响应 P95	个 / 秒
		Spread	spread_pps_http_response	HTTP 响应绝对跨度	个 / 秒
		RSpread	rspread_pps_http_response	HTTP 响应相对跨度	
		StdDev	stddev_packet_http_response	HTTP 响应标准差	个
DNS 请求	dns_request	Sum	sum_packet_dns_request	DNS 请求总量	个

DNS 响应	dns_response	Avg	avg_pps_dns_request	DNS 请求 均值	个 / 秒
		Max	max_pps_dns_request	DNS 请求 峰值	个 / 秒
		Min	min_pps_dns_request	DNS 请求 谷值	个 / 秒
		PCTL	percentile_pps_dns_request	DNS 请求 P95	个 / 秒
		Spread	spread_pps_dns_request	DNS 请求 绝对跨度	个 / 秒
		RSpread	rspread_pps_dns_request	DNS 请求 相对跨度	
		StdDev	stddev_packet_dns_request	DNS 请求 标准差	个
		Sum	sum_packet_dns_response	DNS 响应 总量	个
		Avg	avg_pps_dns_response	DNS 响应 均值	个 / 秒
		Max	max_pps_dns_response	DNS 响应 峰值	个 / 秒
		Min	min_pps_dns_response	DNS 响应 谷值	个 / 秒
		PCTL	percentile_pps_dns_response	DNS 响应 P95	个 / 秒
		Spread	spread_pps_dns_response	DNS 响应 绝对跨度	个 / 秒
		RSpread	rspread_pps_dns_response	DNS 响应 相对跨度	

StdDev stddev_packet_dns_response DNS 响应
标准差 个

负载

传输层负载

指标量	请求字段	算子	响应字段	统计值	单位
活跃连接	flow_load	Avg	avg_l4_fps	活跃连接均值	连接
		Max	max_l4_fps	活跃连接峰值	连接
		Min	min_l4_fps	活跃连接谷值	连接
		PCTL	percentile_l4_fps	活跃连接 P95	连接
		Spread	spread_l4_fps	活跃连接绝对跨度	连接
		RSpread	rspread_l4_fps	活跃连接相对跨度	
		StdDev	stddev_l4_fps	活跃连接标准差	连接

时延

传输层时延

传输层时延统计 TCP 建连和传输时期的网络路径、系统协议栈、上层应用时延，UDP 传输时期的上层应用时延。

指标量	请求字段	算子	响应字段	统计值	单位
平均 TCP 建连时延	rtt	Avg	avg_rtt	平均 TCP 建连时延均值	微秒
		Max	max_rtt	平均 TCP 建连时延峰值	微秒
		Min	min_rtt	平均 TCP 建连时延谷值	微秒
		PCTL	percentile_rtt	平均 TCP 建连时延 P95	微秒
		Spread	spread_rtt	平均 TCP 建连时延绝对跨度	微秒

平均 TCP 建连客户 端时延	rtt_client	RSpread	rspread_rtt	对跨度 平均 TCP 建连时延相 对跨度	
		StdDev	stddev_rtt	平均 TCP 建连时延标 准差	微 秒
		Apdex	stddev_rtt	平均 TCP 建连时延标 准差	微 秒
		Avg	avg_rtt_client	平均 TCP 建连客户端 时延均值	微 秒
		Max	max_rtt_client	平均 TCP 建连客户端 时延峰值	微 秒
		Min	min_rtt_client	平均 TCP 建连客户端 时延谷值	微 秒
		PCTL	percentile_rtt_client	平均 TCP 建连客户端 时延 P95	微 秒
		Spread	spread_rtt_client	平均 TCP 建连客户端 时延绝对跨 度	微 秒
		RSpread	rspread_rtt_client	平均 TCP 建连客户端 时延相对跨 度	
		StdDev	stddev_rtt_client	平均 TCP 建连客户端 时延标准差	微 秒
平均 TCP 建连服务	rtt_server	Apdex	apdex_rtt_client	平均 TCP 建连客户端 时延满意度	%
		Avg	avg_rtt_server	平均 TCP 建连服务端	微 秒

端时延			时延均值	
	Max	max_rtt_server	平均 TCP 建连服务端时延峰值	微秒
	Min	min_rtt_server	平均 TCP 建连服务端时延谷值	微秒
	PCTL	percentile_rtt_server	平均 TCP 建连服务端时延 P95	微秒
	Spread	spread_rtt_server	平均 TCP 建连服务端时延绝对跨度	微秒
	RSpread	rsread_rtt_server	平均 TCP 建连服务端时延相对跨度	
	StdDev	stddev_rtt_server	平均 TCP 建连服务端时延标准差	微秒
	Apdex	apdex_rtt_server	平均 TCP 建连服务端时延满意度	%
平均 TCP 系统时延	Avg	avg_srt	平均 TCP 系统时延均值	微秒
	Max	max_srt	平均 TCP 系统时延峰值	微秒
	Min	min_srt	平均 TCP 系统时延谷值	微秒
	PCTL	percentile_srt	平均 TCP 系统时延 P95	微秒
	Spread	spread_srt	平均 TCP 系统时延绝对	微秒

平均应用时延	art	RSpread	rsread_srt	对跨度 平均 TCP 系统时延相 对跨度	
		StdDev	stddev_srt	平均 TCP 系统时延标 准差	微秒
		Apdex	apdex_srt	平均 TCP 系统时延满 意度	%
		Avg	avg_art	平均应用时 延均值	微秒
		Max	max_art	平均应用时 延峰值	微秒
		Min	min_art	平均应用时 延谷值	微秒
		PCTL	percentile_art	平均应用时 延 P95	微秒
		Spread	spread_art	平均应用时 延绝对跨度	微秒
		RSpread	rsread_art	平均应用时 延相对跨度	
		StdDev	stddev_art	平均应用时 延标准差	微秒
最大 TCP 建连时延	rtt_max	Apdex	apdex_art	平均应用时 延满意度	%
		Avg	avg_rtt_max	最大 TCP 建连时延均 值	微秒
		Max	max_rtt_max	最大 TCP 建连时延峰 值	微秒
		Min	min_rtt_max	最大 TCP 建连时延谷 值	微秒
		PCTL	percentile_rtt_max	最大 TCP 建连时延 P95	微秒

最大 TCP 建连客户 端时延	rtt_client_max	Spread	spread_rtt_max	最大 TCP 建连时延绝 对跨度	微 秒
		RSpread	rspread_rtt_max	最大 TCP 建连时延相 对跨度	
		StdDev	stddev_rtt_max	最大 TCP 建连时延标 准差	微 秒
		Apdex	apdex_rtt_max	最大 TCP 建连时延满 意度	%
		Avg	avg_rtt_client_max	最大 TCP 建连客户端 时延均值	微 秒
		Max	max_rtt_client_max	最大 TCP 建连客户端 时延峰值	微 秒
		Min	min_rtt_client_max	最大 TCP 建连客户端 时延谷值	微 秒
		PCTL	percentile_rtt_client_max	最大 TCP 建连客户端 时延 P95	微 秒
		Spread	spread_rtt_client_max	最大 TCP 建连客户端 时延绝对跨 度	微 秒
		RSpread	rspread_rtt_client_max	最大 TCP 建连客户端 时延相对跨 度	
		StdDev	stddev_rtt_client_max	最大 TCP 建连客户端 时延标准差	微 秒
		Apdex	apdex_rtt_client_max	最大 TCP 建连客户端 时延满意度	%

最大 TCP 建连服务 端时延	rtt_server_max	Avg	avg_rtt_server_max	最大 TCP 建连服务端 时延均值	微秒
		Max	max_rtt_server_max	最大 TCP 建连服务端 时延峰值	微秒
		Min	min_rtt_server_max	最大 TCP 建连服务端 时延谷值	微秒
		PCTL	percentile_rtt_server_max	最大 TCP 建连服务端 时延 P95	微秒
		Spread	spread_rtt_server_max	最大 TCP 建连服务端 时延绝对跨 度	微秒
		RSpread	rsread_rtt_server_max	最大 TCP 建连服务端 时延相对跨 度	
		StdDev	stddev_rtt_server_max	最大 TCP 建连服务端 时延标准差	微秒
		Apdex	apdex_rtt_server_max	最大 TCP 建连服务端 时延满意度	%
最大 TCP 系统时延	srt_max	Avg	avg_srt_max	最大 TCP 系统时延均 值	微秒
		Max	max_srt_max	最大 TCP 系统时延峰 值	微秒
		Min	min_srt_max	最大 TCP 系统时延谷 值	微秒
		PCTL	percentile_srt_max	最大 TCP 系统时延 P95	微秒

最大应用时延	art_max	Spread	spread_srt_max	最大 TCP 系统时延绝对跨度	微秒
		RSpread	rsread_srt_max	最大 TCP 系统时延相对跨度	
		StdDev	stddev_srt_max	最大 TCP 系统时延标准差	微秒
		Apdex	apdex_srt_max	最大 TCP 系统时延满意度	%
		Avg	avg_art_max	最大应用时延均值	微秒
		Max	max_art_max	最大应用时延峰值	微秒
		Min	min_art_max	最大应用时延谷值	微秒
		PCTL	percentile_art_max	最大应用时延 P95	微秒
		Spread	spread_art_max	最大应用时延绝对跨度	微秒
		RSpread	rsread_art_max	最大应用时延相对跨度	
		StdDev	stddev_art_max	最大应用时延标准差	微秒
		Apdex	apdex_art_max	最大应用时延满意度	%

应用层时延

统计 HTTP、DNS 的请求和响应间的时间间隔。

指标量	请求字段	算子	响应字段	统计值	单位
平均 HTTP 时延	http_rrt	Avg	avg_http_rrt	平均 HTTP 时延均值	微秒
		Max	max_http_rrt	平均 HTTP 时	微

平均 DNS 时延	dns_rrt	Min	min_http_rrt	延峰值	秒
				平均 HTTP 时延谷值	微秒
		PCTL	percentile_http_rrt	平均 HTTP 时延 P95	微秒
		Spread	spread_http_rrt	平均 HTTP 时延绝对跨度	微秒
		RSpread	rspread_http_rrt	平均 HTTP 时延相对跨度	
		StdDev	stddev_http_rrt	平均 HTTP 时延标准差	微秒
		Apdex	apdex_http_rrt	平均 HTTP 时延满意度	%
		Avg	avg_dns_rrt	平均 DNS 时延均值	微秒
		Max	max_dns_rrt	平均 DNS 时延峰值	微秒
		Min	min_dns_rrt	平均 DNS 时延谷值	微秒
		PCTL	percentile_dns_rrt	平均 DNS 时延 P95	微秒
		Spread	spread_dns_rrt	平均 DNS 时延绝对跨度	微秒
		RSpread	rspread_dns_rrt	平均 DNS 时延相对跨度	
		StdDev	stddev_dns_rrt	平均 DNS 时延标准差	微秒
		Apdex	apdex_dns_rrt	平均 DNS 时延满意度	%
最大 HTTP 时延	http_rrt_max	Avg	avg_http_rrt_max	最大 HTTP 时延均值	微秒
		Max	max_http_rrt_max	最大 HTTP 时延峰值	微秒
		Min	min_http_rrt_max	最大 HTTP 时延谷值	微秒
		PCTL	percentile_http_rrt_max	最大 HTTP 时	微

最大 DNS 时延	dns_rrt_max			延 P95	秒
		Spread	spread_http_rrt_max	最大 HTTP 时延绝对跨度	微秒
		RSpread	rspread_http_rrt_max	最大 HTTP 时延相对跨度	
		StdDev	stddev_http_rrt_max	最大 HTTP 时延标准差	微秒
		Apdex	apdex_http_rrt_max	最大 HTTP 时延满意度	%
		Avg	avg_dns_rrt_max	最大 DNS 时延均值	微秒
		Max	max_dns_rrt_max	最大 DNS 时延峰值	微秒
		Min	min_dns_rrt_max	最大 DNS 时延谷值	微秒
		PCTL	percentile_dns_rrt_max	最大 DNS 时延 P95	微秒
		Spread	spread_dns_rrt_max	最大 DNS 时延绝对跨度	微秒
		RSpread	rspread_dns_rrt_max	最大 DNS 时延相对跨度	
		StdDev	stddev_dns_rrt_max	最大 DNS 时延标准差	微秒
		Apdex	apdex_dns_rrt_max	最大 DNS 时延满意度	%

性能

传输层性能

传输层性能统计 TCP 重传与零窗。

指标量	请求字段	算子	响应字段	统计值	单位
TCP 重传	retrans	Sum	sum_retrans	TCP 重传总量	包
		Avg	avg_rps	TCP 重传	包 /

TCP 客户 端重 传	retrans_tx			均值	秒
		Max	max_rps	TCP 重传 峰值	包 / 秒
		Min	min_rps	TCP 重传 谷值	包 / 秒
		PCTL	percentile_rps	TCP 重传 P95	包 / 秒
		Spread	spread_rps	TCP 重传 绝对 跨度	包 / 秒
		RSpread	rsread_rps	TCP 重传 相对 跨度	
		StdDev	stddev_retrans	TCP 重传 标准 差	包
		Sum	sum_retrans_tx	TCP 客户 端重 传总 量	包
		Avg	avg_rps_tx	TCP 客户 端重 传均 值	包 / 秒
		Max	max_rps_tx	TCP 客户 端重 传峰 值	包 / 秒
		Min	min_rps_tx	TCP 客户 端重	包 / 秒

TCP 服务端重 传	retrans_rx	PCTL	percentile_rps_tx	传谷 值 TCP 客户 端重 传 P95	包 / 秒
		Spread	spread_rps_tx	TCP 客户 端重 传绝 对跨 度	包 / 秒
		RSpread	rsread_rps_tx	TCP 客户 端重 传相 对跨 度	
		StdDev	stddev_retrans_tx	TCP 客户 端重 传标 准差	包
		Sum	sum_retrans_rx	TCP 服务 端重 传总 量	包
		Avg	avg_rps_rx	TCP 服务 端重 传均 值	包 / 秒
		Max	max_rps_rx	TCP 服务 端重 传峰 值	包 / 秒
		Min	min_rps_rx	TCP	包

TCP 零窗	zero_win			服务端重传谷值	/秒
		PCTL	percentile_rps_rx	TCP服务端重传P95	包/秒
		Spread	spread_rps_rx	TCP服务端重传绝对跨度	包/秒
		RSpread	rsread_rps_rx	TCP服务端重传相对跨度	
		StdDev	stddev_retrans_rx	TCP服务端重传标准差	包
		Sum	sum_zero_win	TCP零窗总量	包
		Avg	avg_zwps	TCP零窗均值	包/秒
		Max	max_zwps	TCP零窗峰值	包/秒
		Min	min_zwps	TCP零窗谷值	包/秒
		PCTL	percentile_zwps	TCP零窗	包/

TCP 客户 端零 窗	zero_win_tx	Spread	spread_zwps	P95 TCP 零窗 绝对 跨度	秒 包 / 秒
		RSpread	rspread_zwps	TCP 零窗 相对 跨度	
		StdDev	stddev_zero_win	TCP 零窗 标准 差	包
		Sum	sum_zero_win_tx	TCP 客户 端零 窗总 量	包
		Avg	avg_zwps_tx	TCP 客户 端零 窗均 值	包 / 秒
		Max	max_zwps_tx	TCP 客户 端零 窗峰 值	包 / 秒
		Min	min_zwps_tx	TCP 客户 端零 窗谷 值	包 / 秒
		PCTL	percentile_zwps_tx	TCP 客户 端零 窗 P95	包 / 秒
		Spread	spread_zwps_tx	TCP 客户	包 /

TCP 服务 端零 窗	zero_win_rx	RSpread	rspread_zwps_tx	TCP 客户 端零 窗相 对跨 度	秒
		StdDev	stddev_zero_win_tx	TCP 客户 端零 窗标 准差	包
		Sum	sum_zero_win_rx	TCP 服务 端零 窗总 量	包
		Avg	avg_zwps_rx	TCP 服务 端零 窗均 值	包 / 秒
		Max	max_zwps_rx	TCP 服务 端零 窗峰 值	包 / 秒
		Min	min_zwps_rx	TCP 服务 端零 窗谷 值	包 / 秒
		PCTL	percentile_zwps_rx	TCP 服务 端零 窗 P95	包 / 秒

TCP 重传 比例	retrans_ratio	Spread	spread_zwps_rx	TCP 服务端零窗绝对跨度	包 / 秒
		RSpread	rsread_zwps_rx	TCP 服务端零窗相对跨度	
		StdDev	stddev_zero_win_rx	TCP 服务端零窗标准差	包
		Sum	sum_avg_retrans_ratio	累计 TCP 重传比例	%
		Avg	avg_avg_retrans_ratio	TCP 重传比例均值	%
		Max	max_avg_retrans_ratio	TCP 重传比例峰值	%
		Min	min_avg_retrans_ratio	TCP 重传比例谷值	%
		PCTL	percentile_avg_retrans_ratio	TCP 重传比例 P95	%
		Spread	spread_retrans_ratio	TCP 重传比例	%

TCP 客户 端重 传比 例	retrans_tx_ratio	RSpread	rspread_retrans_ratio	绝对 跨度 TCP 重传 比例 绝对 跨度	
		StdDev	stddev_avg_retrans_ratio	TCP 重传 比例 标准 差	%
		Sum	sum_avg_retrans_tx_ratio	累计 TCP 客户 端重 传比 例	%
		Avg	avg_avg_retrans_tx_ratio	TCP 客户 端重 传比 例均 值	%
		Max	max_avg_retrans_tx_ratio	TCP 客户 端重 传比 例峰 值	%
		Min	min_avg_retrans_tx_ratio	TCP 客户 端重 传比 例谷 值	%
		PCTL	percentile_avg_retrans_tx_ratio	TCP 客户 端重 传比	%

TCP 服务端重 传比 例	retrans_rx_ratio	Spread	spread_retrans_tx_ratio	例 P95 TCP 客户 端重 传比 例绝 对跨 度	%
		RSpread	rspread_retrans_tx_ratio	TCP 客户 端重 传比 例绝 对跨 度	
		StdDev	stddev_avg_retrans_tx_ratio	TCP 客户 端重 传比 例标 准差	%
		Sum	sum_avg_retrans_tx_ratio	累计 TCP 服务 端重 传比 例	%
		Avg	avg_avg_retrans_tx_ratio	TCP 服务 端重 传比 例均 值	%
		Max	max_avg_retrans_tx_ratio	TCP 服务 端重 传比 例峰 值	%

		Min	min_avg_retrans_tx_ratio	TCP 服务端重 传比 例谷 值	%
		PCTL	percentile_avg_retrans_tx_ratio	TCP 服务端重 传 P95	%
		Spread	spread_retrans_rx_ratio	TCP 服务端重 传比 例绝 对跨 度	%
		RSpread	rsread_retrans_rx_ratio	TCP 服务端重 传比 例绝 对跨 度	
		StdDev	stddev_avg_retrans_rx_ratio	TCP 服务端重 传比 例标 准差	%
TCP 零窗 比例	zero_win_ratio	Sum	sum_avg_zero_win_ratio	累计 TCP 零窗 比例	%
		Avg	avg_avg_zero_win_ratio	TCP 零窗 比例 均值	%
		Max	max_avg_zero_win_ratio	TCP	%

TCP 客户 端零 窗比 例	zero_win_tx_ratio	Min	max_avg_zero_win_ratio	零窗 比例 峰值 TCP 零窗 比例 谷值	%
		PCTL	percentile_avg_zero_win_ratio	TCP 零窗 比例 P95	%
		Spread	spread_zero_win_ratio	TCP 零窗 比例 绝对 跨度	%
		RSpread	rspread_zero_win_ratio	TCP 零窗 比例 绝对 跨度	
		StdDev	stddev_avg_zero_win_ratio	TCP 零窗 比例 标准 差	%
		Sum	sum_avg_zero_win_tx_ratio	累计 TCP 客户 端零 窗比 例	%
		Avg	avg_avg_zero_win_tx_ratio	TCP 客户 端零 窗比 例均 值	%
		Max	max_avg_zero_win_tx_ratio	TCP 客户	%

					端零窗比例峰值	
					TCP客户端零窗比例谷值	%
					TCP客户端零窗比例P95	%
			Min	max_avg_zero_win_tx_ratio	TCP客户端零窗比例谷值	%
			PCTL	percentile_avg_zero_win_tx_ratio	TCP客户端零窗比例P95	%
			Spread	spread_zero_win_tx_ratio	TCP客户端零窗比例绝对跨度	%
			RSpread	rspread_zero_win_tx_ratio	TCP客户端零窗比例绝对跨度	
			StdDev	stddev_avg_zero_win_tx_ratio	TCP客户端零窗比例标准差	%
TCP服务端零窗比	zero_win_rx_ratio	Sum	sum_avg_zero_win_rx_ratio	累计TCP服务端零		%

例

			窗比 例	
Avg	avg_avg_zero_win_rx_ratio	TCP 服务 端零 窗比 例均 值	%	
Max	max_avg_zero_win_rx_ratio	TCP 服务 端零 窗比 例峰 值	%	
Min	max_avg_zero_win_rx_ratio	TCP 服务 端零 窗比 例谷 值	%	
PCTL	percentile_avg_zero_win_rx_ratio	TCP 服务 端零 窗 P95	%	
Spread	spread_zero_win_rx_ratio	TCP 服务 端零 窗比 例绝 对跨 度	%	
RSpread	rspread_zero_win_rx_ratio	TCP 服务 端零 窗比 例绝 对跨 度		
StdDev	stddev_avg_zero_win_rx_ratio	TCP	%	

服务端
窗口比例
标准差

异常

传输层异常

传输层异常统计异常的 TCP 建连和断连数量。

指标量	请求参数	算子	响应参数	统计值	单位
建连-失败次数	tcp_establish_fail	Sum	sum_tcp_establish_fail	建连-失败总量	连接
		Avg	avg_tcp_efps	建连-失败均值	连接/秒
		Max	max_tcp_efps	建连-失败峰值	连接/秒
		Min	min_tcp_efps	建连-失败谷值	连接/秒
		PCTL	percentile_tcp_efps	建连-	连接

				失败 P95	/秒
		Spread	spread_tcp_efps	建连-失败绝对跨度	连接/秒
		RSpread	rspread_tcp_efps	建连-失败相对跨度	
		StdDev	stddev_tcp_establish_fail	建连-失败标准差	连接
建连-客户端失败次数	client_establish_fail	Sum	sum_client_establish_fail	建连-客户端失败总量	连接
		Avg	avg_client_efps	建连-客户	连接/秒

		端失败均值	
Max	max_client_efps	建连-客户	连接 / 秒
		端失败峰值	
Min	min_client_efps	建连-客户	连接 / 秒
		端失败谷值	
PCTL	percentile_client_tcp_efps	建连-客户	连接 / 秒
		端失败P95	
Spread	spread_client_efps	建连-客户	连接 / 秒
		端失败绝	

					对 跨 度 建 连- 客 户 端 失 败 相 对 跨 度 建 连- 客 户 端 失 败 标 准 差 建 连- 服 务 端 失 败 总 量 建 连- 服 务 端 失 败
			RSprea d	rspread_client_efps	
			StdDev	stddev_client_establish_fail	连 接
建 连- 服 务 端 失 败 次 数	server_establish_fail	Sum	sum_server_establish_fail		连 接
		Avg	avg_server_efps		连 接 / 秒

Max	max_server_efps	均 值 建 连- 服 务 端 失 败 峰 值	连 接 / 秒
Min	min_server_efps	建 连- 服 务 端 失 败 谷 值	连 接 / 秒
PCTL	percentile_server_tcp_efps	建 连- 服 务 端 失 败 P9 5	连 接 / 秒
Spread	spread_server_efps	建 连- 服 务 端 失 败 绝 对 跨 度	连 接 / 秒

		RSpread	rspread_server_efps	建连-服务端失败相对跨度	
		StdDev	stddev_server_establish_fail	建连-服务端失败标准差	连接
建连-失败比例	tcp_establish_fail_ratio	Sum	sum_avg_tcp_establish_fail_ratio	建连-累计失败比例	%
		Avg	avg_avg_tcp_establish_fail_ratio	建连-失败比例均值	%
		Max	max_avg_tcp_establish_fail_ratio	建连-失	%

		败 比 例 峰 值	
Min	min_avg_tcp_establish_fail_ratio	建 连- 失 败 比 例 谷 值	%
PCTL	percentile_avg_tcp_establish_fail_ratio	建 连- 失 败 比 例 P9 5	%
Spread	spread_tcp_establish_fail_ratio	建 连- 失 败 比 例 绝 对 跨 度	%
RSprea d	rspread_tcp_establish_fail_ratio	建 连- 失 败 比 例 相 对 跨	

建连-客户端失败比例	client_establish_fail_ratio	StdDev	stddev_avg_tcp_establish_fail_ratio	度建连-失败比例标准差	%
		Sum	sum_avg_client_establish_fail_ratio	建连-累计客户端失败比例	%
		Avg	avg_avg_client_establish_fail_ratio	建连-客户端失败比例均值	%
		Max	max_avg_client_establish_fail_ratio	建连-客户端失败比	%

Min	min_avg_client_establish_fail_ratio	例峰值 值 建 连- 客 户 端 失 败 比 例 谷 值	%
PCTL	percentile_avg_client_establish_fail_ratio	建 连- 客 户 端 失 败 比 例 P9 5	%
Spread	spread_client_establish_fail_ratio	建 连- 客 户 端 失 败 比 例 绝 对 跨 度	%
RSprea d	rspread_client_establish_fail_ratio	建 连-	

		败 比 例 均 值	
Max	max_avg_server_establish_fail_ratio	建连-服务端失败比例峰值	%
Min	min_avg_server_establish_fail_ratio	建连-服务端失败比例谷值	%
PCTL	percentile_avg_server_establish_fail_ratio	建连-服务端失败比例P95	%
Spread	spread_server_establish_fail_ratio	建连-	%

				服务端失败比例绝对跨度	
		RSpread	rsread_server_establish_fail_ratio	建连-服务端失败比例相对跨度	
		StdDev	stddev_avg_server_establish_fail_ratio	建连-服务端失败比例标准差	%
建连-客户	client_source_port_reuse	Sum	sum_client_source_port_reuse	建连-客户	连接

端
端
口
复
用

Avg avg_client_sprps

Max max_client_sprps

Min min_client_sprps

端
端
口
复
用
总
量
建
连
-
客
户
端
端
口
复
用
均
值
建
连
-
客
户
端
端
口
复
用
峰
值
建
连
-
客
户
端
端
口
复
用
谷
值

连
接
/
秒

连
接
/
秒

连
接
/
秒

PCTL	percentile_sprps	建连-客户端口复用P95	连接/秒
Spread	spread_client_sprps	建连-客户端口复用绝对跨度	连接/秒
RSprea d	rsread_client_sprps	建连-客户端口复用相对跨度	
StdDev	stddev_client_source_port_reuse	建连-客户	连接

建连-客户端SYN结束	client_syn_repeat	Sum	sum_client_syn_repeat	端口复用标准差	连接
				建连-客户端SYN结束总量	
		Avg	avg_client_srps	建连-客户端SYN结束均值	连接 / 秒
		Max	max_client_srps	建连-客户端SYN结束峰值	连接 / 秒

Min	min_client_srps	建连-客户端SYN结束谷值	连接/秒
PCTL	percentile_client_srps	建连-客户端SYN结束P95	连接/秒
Spread	spread_client_srps	建连-客户端SYN结束绝对跨度	连接/秒
RSprea d	rspread_client_srps	建连-客户端SYN	

				N 结束相对跨度	
				建连-客户端SYN结束标准差	连接
		StdDev	stddev_client_syn_repeat		
建连-客户端其他重置	client_establish_other_rst	Sum	sum_client_establish_other_rst	建连-客户端其他重置总量	连接
		Avg	avg_client_eorps	建连-客户端其他重置均值	连接 / 秒

Max	max_client_eorps	建连-客户端其他重置峰值	连接 / 秒
Min	min_client_eorps	建连-客户端其他重置谷值	连接 / 秒
PCTL	percentile_client_eorps	建连-客户端其他重置P95	连接 / 秒
Spread	spread_client_eorps	建连-客户端其他重	连接 / 秒

					置绝对跨度建连-客户端其他重置相对跨度建连-客户端其他重置标准差建连-服务端直接重置总
		RSprea d	rspread_client_eorps		
		StdDev	stddev_client_establish_other_rst		连接
建连-服务端直接重置	server_reset	Sum	sum_server_reset		连接

Avg	avg_server_rps	量 建 连- 服 务 端 直 接 重 置 均 值	连 接 / 秒
Max	max_server_rps	建 连- 服 务 端 直 接 重 置 峰 值	连 接 / 秒
Min	min_server_rps	建 连- 服 务 端 直 接 重 置 谷 值	连 接 / 秒
PCTL	percentile_server_rps	建 连- 服 务 端 直	连 接 / 秒

Spread spread_server_rps

RSprea rsread_server_rps
d

StdDev stddev_server_reset

接重置
P9
5
建连-服务端直接重置绝对跨度建连-服务端直接重置相对跨度建连-服务端直接重置标
连接 / 秒
连接

建连-服务端SYN结束

server_syn_ack_repea
t

Sum

sum_server_sarps

Avg

avg_server_sarps

Max

max_server_sarps

Min

min_server_sarps

准差建连-服务端SYN结束总量建连-服务端SYN结束均值建连-服务端SYN结束峰值建连-服务端SYN

连接

连接 / 秒

连接 / 秒

连接 / 秒

PCTL	percentile_server_sarps	N 结束 建连-服务端 SYN 结束 P95	连接 / 秒
Spread	spread_server_sarps	建连-服务端 SYN 结束 绝对跨度	连接 / 秒
RSpread	rsread_server_sarps	建连-服务端 SYN 结束 相对跨	

		StdDev	stddev_server_sarps	度 建 连- 服 务 端 SY N 结 束 标 准 差	连 接
建 连- 服 务 端 其 他 重 置	server_establish_othe r_rst	Sum	sum_server_establish_other_rst	建 连- 服 务 端 建 连 其 他 重 置 总 量	连 接
		Avg	avg_server_eorps	建 连- 服 务 端 建 连 其 他 重 置 均 值	连 接 / 秒
		Max	max_server_eorps	建	连

		连-接 / 秒
		服务端建连其他重置峰值
Min	min_server_eorps	建连-连接 / 秒
		服务端建连其他重置谷值
PCTL	percentile_server_eorps	建连-连接 / 秒
		服务端建连其他重置
		P95
Spread	spread_server_eorps	建连-连接 / 服

				务端其他重置绝对跨度建连-服务端其他重置相对跨度建连-服务端其他重置标准差传输-客户端	秒
		RSpread	rsread_server_eorps		
		d			
		StdDev	stddev_server_establish_other_rst	建连-服务端其他重置标准差	连接
传输-客户端	client_rst_flow	Sum	sum_client_rst_flow	传输-客户端	连接

重置

Avg avg_client_rst_fps

Max max_client_rst_fps

Min min_client_rst_fps

PCTL percentile_client_rst_fps

重置总量
传输-客户端重置均值
传输-客户端重置峰值
传输-客户端重置谷值
传输-客户端重置P95
连接 / 秒
连接 / 秒
连接 / 秒
连接 / 秒

	Spread	spread_client_rst_fps	传输-客户端重置绝对跨度	连接 / 秒
	RSpread	rsread_client_rst_fps	传输-客户端重置相对跨度	
	StdDev	stddev_client_rst_flow	传输-客户端重置标准差	连接
传输-服务端队列溢	server_queue_lack	Sum	sum_server_queue_lack	连接

出

Avg	avg_server_qlps
Max	max_server_qlps
Min	min_server_qlps
PCTL	percentile_server_qlps

出总量
传输-服务端队列溢出均值
传输-服务端队列溢出峰值
传输-服务端队列溢出谷值
连接 / 秒
连接 / 秒
连接 / 秒
连接 / 秒

		端 队 列 溢 出 P9 5	连 接 / 秒
Spread	spread_server_qlps	传 输- 服 务 端 队 列 溢 出 绝 对 跨 度	
RSprea d	rsread_server_qlps	传 输- 服 务 端 队 列 溢 出 相 对 跨 度	
StdDev	stddev_server_queue_lack	传 输- 服 务 端 队 列 溢	连 接

传输-
服务端
重置

server_rst_flow

Sum

sum_server_rst_flow

Avg

avg_server_rst_fps

Max

max_server_rst_fps

Min

min_server_rst_fps

出
标
准
差
传
输-
服
务
端
重
置
总
量
传
输-
服
务
端
重
置
均
值
传
输-
服
务
端
重
置
峰
值
传
输-
服
务
端
重
置
谷
值
连
接
连
接
/
秒
连
接
/
秒
连
接
/
秒

PCTL	percentile_server_rst_fps	传输-服务端重置P95	连接 / 秒
Spread	spread_server_rst_fps	传输-服务端重置绝对跨度	连接 / 秒
RSprea d	rspread_server_rst_fps	传输-服务端重置相对跨度	
StdDev	stddev_server_rst_flow	传输-服务端重置标准	连接

断连-客户端半关	client_half_close_flow	Sum	sum_client_half_close_flow	差断连-客户端半关总量	连接
		Avg	avg_client_half_close_fps	断连-客户端半关均值	连接 / 秒
		Max	max_client_half_close_fps	断连-客户端半关峰值	连接 / 秒
		Min	min_client_half_close_fps	断连-客户端半关谷值	连接 / 秒
		PCTL	percentile_client_half_close_fps	断连-客	连接 / 客

				户 端 半 关 P9 5	秒
		Spread	spread_client_half_close_fps	断 连- 客 户 端 半 关 绝 对 跨 度	连 接 / 秒
		RSprea d	rsread_client_half_close_fps	断 连- 客 户 端 半 关 相 对 跨 度	
		StdDev	stddev_client_half_close_flow	断 连- 客 户 端 半 关 标 准 差	连 接
断 连-	server_half_close_flow	Sum	sum_server_half_close_flow	断 连-	连 接

服务端半关

Avg	avg_server_half_close_fps	服务端半关总量 断连-服务端半关均值	连接 / 秒
Max	max_server_half_close_fps	断连-服务端半关峰值	连接 / 秒
Min	min_server_half_close_fps	断连-服务端半关谷值	连接 / 秒
PCTL	percentile_server_half_close_fps	断连-服务端半	连接 / 秒

				关 P9 5	断连-服务端半关绝对跨度断连-服务端半关相对跨度断连-服务端半关标准差断连-TCP连接	连接/秒
Spread		spread_server_half_close_fps				
RSpread		rsread_server_half_close_fps		d		
StdDev		stddev_server_half_close_flow			断连-服务端半关标准差断连-TCP连接	连接
断连-TCP连接	tcp_timeout	Sum	sum_tcp_timeout		断连-TCP连接	连接

超时

Avg avg_tcp_timeout_fps

超时总量
断连-TC
P 连接
超时均
值
断连-TC
P 连接
超时峰
值
断连-TC
P 连接
超
时
谷
值

连接 / 秒

Max max_tcp_timeout_fps

连接 / 秒

Min min_tcp_timeout_fps

连接 / 秒

PCTL percentile_tcp_timeout_fps

连接 / 秒

				时 P9 5	断 连- TCP 连接 超时 绝对 跨度 断 连- TCP 连接 超时 相对 跨度 断 连- TCP 连接 超时 标准 差	连 接 / 秒	
	Spread	spread_tcp_timeout_fps					
	RSpread	rsread_tcp_timeout_fps					
	StdDev	stddev_tcp_timeout			断 连- TCP 连接 超时 标准 差	连 接	
应用层异常							
指标	请求字段	算子	响应字段	统计	单		

量				值	位
HTTP 异常	http_error	Sum	sum_http_error	HTTP 异常总量	个
		Avg	avg_http_eps	HTTP 异常均值	个 / 秒
		Max	max_http_eps	HTTP 异常峰值	个 / 秒
		Min	min_http_eps	HTTP 异常谷值	个 / 秒
		PCTL	percentile_http_eps	HTTP 异常 P95	个 / 秒
		Spread	spread_http_eps	HTTP 异常绝对跨度	个 / 秒
		RSpread	rsread_http_eps	HTTP 异常相对跨度	
		StdDev	stddev_http_error	HTTP 异常标准差	个
HTTP 4XX 异常	http_client_error	Sum	sum_http_client_error	HTTP 4XX 异常总量	个

		Avg	avg_http_client_eps	HTTP 4XX 异常均值	个 / 秒
		Max	max_http_client_eps	HTTP 4XX 异常峰值	个 / 秒
		Min	min_http_client_eps	HTTP 4XX 异常谷值	个 / 秒
		PCTL	percentile_http_client_eps	HTTP 4XX 异常 P95	个 / 秒
		Spread	spread_http_client_eps	HTTP 4XX 异常绝对跨度	个 / 秒
		RSpread	rsread_http_client_eps	HTTP 4XX 异常相对跨度	
		StdDev	stddev_http_client_error	HTTP 4XX 异常标准差	个
HTTP 5XX 异常	http_server_error	Sum	sum_http_server_error	HTTP 5XX 异常	个

				总量	
HTTP 服	Avg	avg_http_server_eps	HTTP 5XX 异常均值	个 / 秒	
	Max	max_http_server_eps	HTTP 5XX 异常峰值	个 / 秒	
	Min	min_http_server_eps	HTTP 5XX 异常谷值	个 / 秒	
	PCTL	percentile_http_server_eps	HTTP 5XX 异常 P95	个 / 秒	
	Spread	spread_http_server_eps	HTTP 5XX 异常绝对跨度	个 / 秒	
	RSprea d	rsread_http_server_eps	HTTP 5XX 异常相对跨度		
	StdDev	stddev_http_server_error	HTTP 5XX 异常标准差	个	
HTTP 服	http_timeout	Sum	sum_http_timeout	HTTP 服	个

务端 超时			务端 超时 总量	
	Avg	avg_http_timetout_eps	HTT P 服 务端 超时 均值	个 / 秒
	Max	max_http_timetout_eps	HTT P 服 务端 超时 峰值	个 / 秒
	Min	min_http_timetout_eps	HTT P 服 务端 超时 谷值	个 / 秒
	PCTL	percentile_http_timetout_eps	HTT P 服 务端 超时 P95	个 / 秒
	Spread	spread_http_timeout_eps	HTT P 服 务端 超时 绝对 跨度	个 / 秒
	RSprea d	rspread_http_timeout_eps	HTT P 服 务端 超时 相对 跨度	
	StdDev	stddev_http_timeout	HTT P 服 务端 超时 标准	个

DNS 异常	dns_error	Sum	sum_dns_error	差	DNS 异常总量	个
		Avg	avg_dns_eps		DNS 异常均值	个 / 秒
		Max	max_dns_eps		DNS 异常峰值	个 / 秒
		Min	min_dns_eps		DNS 异常谷值	个 / 秒
		PCTL	percentile_dns_eps		DNS 异常 P95	个 / 秒
		Spread	spread_dns_eps		DNS 异常绝对跨度	个 / 秒
		RSprea d	rsread_dns_eps		DNS 异常相对跨度	
		StdDev	stddev_dns_error		DNS 异常标准差	个
DNS 客户 端异 常	dns_client_error	Sum	sum_dns_client_error		DNS 客户端异常总量	个
		Avg	avg_dns_client_eps		DNS 客户端异常均值	个 / 秒
		Max	max_dns_client_eps		DNS 客户	个 /

DNS 服务 端异 常	dns_server_error			端异 常峰 值	秒
		Min	min_dns_client_eps	DNS 客户 端异 常谷 值	个 / 秒
		PCTL	percentile_dns_client_eps	DNS 客户 端异 常 P95	个 / 秒
		Spread	spread_dns_client_eps	DNS 客户 端异 常绝 对跨 度	个 / 秒
		RSprea d	rsread_dns_client_eps	DNS 客户 端异 常相 对跨 度	
		StdDev	stddev_dns_client_error	DNS 客户 端异 常标 准差	个
DNS 服务 端异 常	dns_server_error	Sum	sum_dns_server_error	DNS 服务 端异 常总 量	个
		Avg	avg_dns_server_eps	DNS 服务 端异 常均 值	个 / 秒

		Max	max_dns_server_eps	DNS 服务 端异 常峰 值	个 / 秒
		Min	min_dns_server_eps	DNS 服务 端异 常谷 值	个 / 秒
		PCTL	percentile_dns_server_eps	DNS 服务 端异 常 P95	个 / 秒
		Spread	spread_dns_server_eps	DNS 服务 端异 常绝 对跨 度	个 / 秒
		RSprea d	rsread_dns_server_eps	DNS 服务 端异 常相 对跨 度	
		StdDev	stddev_dns_server_error	DNS 服务 端异 常标 准差	个
DNS 服务 端超 时	dns_timeout	Sum	sum_dns_timeout	DNS 服务 端超 时总 量	个
		Avg	avg_dns_timetout_eps	DNS 服务 端超	个 / 秒

				时均 值	
		Max	max_dns_timeout_eps	DNS 服务 端超 时峰 值	个 / 秒
		Min	min_dns_timeout_eps	DNS 服务 端超 时谷 值	个 / 秒
		PCTL	percentile_dns_timeout_eps	DNS 服务 端超 时 P95	个 / 秒
		Spread	spread_dns_timeout_eps	DNS 服务 端超 时绝 对跨 度	个 / 秒
		RSprea d	rsread_dns_timeout_eps	DNS 服务 端超 时相 对跨 度	
HTT P 异 常比 例	http_error_ratio	StdDev	stddev_dns_timeout	DNS 服务 端超 时标 准差	个
		Sum	sum_avg_http_error_ratio	累计 HTT P 异 常比 例	%
		Avg	avg_avg_http_error_ratio	HTT	%

				P 异常比例均值	
	Max	max_avg_http_error_ratio	HTTP 异常比例峰值	%	
	Min	min_avg_http_error_ratio	HTTP 异常比例谷值	%	
	PCTL	percentile_avg_http_error_ratio	HTTP 异常比例 P95	%	
	Spread	spread_http_error_ratio	HTTP 异常比例绝对跨度	%	
	RSpread	rspread_http_error_ratio	HTTP 异常比例相对跨度		
	StdDev	stddev_avg_http_error_ratio	HTTP 异常比例标准差	%	
HTTP 4XX 比例	http_client_error_ratio	Sum	sum_avg_http_client_error_ratio	累计 HTTP 4XX	%

				异常比例	
		Avg	avg_avg_http_client_error_ratio	HTTP 4XX 比例均值	%
		Max	max_avg_http_client_error_ratio	HTTP 4XX 比例峰值	%
		Min	min_avg_http_client_error_ratio	HTTP 4XX 比例谷值	%
		PCTL	percentile_avg_http_client_error_ratio	HTTP 4XX 比例 P95	%
		Spread	spread_http_client_error_ratio	HTTP 4XX 比例绝对跨度	%
		RSpread	rsread_http_client_error_ratio	HTTP 4XX 比例相对跨度	
		StdDev	stddev_avg_http_client_error_ratio	HTTP 4XX 比例标准差	%
HTTP	http_server_error_ratio	Sum	sum_avg_http_server_error_ratio	累计	%

5XX 比例	atio			HTT P 5XX 异常 比例	
Avg	avg_avg_http_server_error_ratio			HTT P 5XX 比例 均值	%
Max	max_avg_http_server_error_ratio			HTT P 5XX 比例 峰值	%
Min	min_avg_http_server_error_ratio			HTT P 5XX 比例 谷值	%
PCTL	percentile_avg_http_server_error_ratio			HTT P 5XX 比例 P95	%
Spread	spread_http_server_error_ratio			HTT P 5XX 比例 绝对 跨度	%
RSprea d	rspread_http_server_error_ratio			HTT P 5XX 比例 相对 跨度	
StdDev	stddev_avg_http_server_error_ratio			HTT P 5XX 比例 标准	%

DNS 异常 比例	dns_error_ratio	Sum	sum_avg_dns_error_ratio	差 累计 DNS 异常 比例	%
		Avg	avg_avg_dns_error_ratio	DNS 异常 比例 均值	%
		Max	max_avg_dns_error_ratio	DNS 异常 比例 峰值	%
		Min	min_avg_dns_error_ratio	DNS 异常 比例 谷值	%
		PCTL	percentile_avg_dns_error_ratio	DNS 异常 比例 P95	%
		Spread	spread_dns_error_ratio	DNS 异常 比例 绝对 跨度	%
		RSprea d	rsread_dns_error_ratio	DNS 异常 比例 相对 跨度	
		StdDev	stddev_avg_dns_error_ratio	DNS 异常 比例 标准 差	%
DNS 客户 端异 常比	dns_client_error_rat io	Sum	sum_avg_dns_client_error_ratio	累计 DNS 客户 端异	%

例

			常比例	
Avg	avg_avg_dns_client_error_ratio	DNS 客户端异常比例均值	%	
Max	max_avg_dns_client_error_ratio	DNS 客户端异常比例峰值	%	
Min	min_avg_dns_client_error_ratio	DNS 客户端异常比例谷值	%	
PCTL	percentile_avg_dns_client_error_ratio	DNS 客户端异常比例 P95	%	
Spread	spread_dns_client_error_ratio	DNS 客户端异常比例绝对跨度	%	
RSprea d	rspread_dns_client_error_ratio	DNS 客户端异常比例相对跨度		

DNS 服务 端异 常比 例	dns_server_error_ratio	StdDev	stddev_avg_dns_client_error_ratio	DNS 客 户 端 异 常 比 例 标 准 差	%
		Sum	sum_avg_dns_server_error_ratio	累 计 D N S 服 务 端 异 常 比 例	%
		Avg	avg_avg_dns_server_error_ratio	D N S 服 务 端 异 常 比 例 均 值	%
		Max	max_avg_dns_server_error_ratio	D N S 服 务 端 异 常 比 例 峰 值	%
		Min	min_avg_dns_server_error_ratio	D N S 服 务 端 异 常 比 例 谷 值	%
		PCTL	percentile_avg_dns_server_error_ratio	D N S 服 务 端 异 常 比 例 P95	%
		Spread	spread_dns_server_error_ratio	D N S 服 务 端 异 常 比	%

					例绝对跨度
		RSpread	rspread_dns_server_error_ratio		DNS服务端异常比例相对跨度
		StdDev	stddev_avg_dns_server_error_ratio		DNS服务端异常比例标准差
					%

属性

资源属性

指标量	请求字段	算子	响应字段	统计值	单位
VPC	l3_epc_id	Distinct	distinct_l3_epc_id	VPC 个数	个
子网	subnet_id	Distinct	distinct_subnet_id	子网个数	个
IP 地址	ip	Distinct	distinct_ip	IP 地址个数	个
容器节点	pod_node_id	Distinct	distinct_pod_node_id	容器节点个数	个
命名空间	pod_ns_id	Distinct	distinct_pod_ns_id	命名空间个数	个
工作负载	pod_group_id	Distinct	distinct_pod_group_id	工作负载个数	个
POD	pod_id	Distinct	distinct_pod_id	POD 个数	个
宿主机	host_id	Distinct	distinct_host_id	宿主机个数	个
云服务器	vm_id	Distinct	distinct_vm_id	云服务器个数	个
区域	region_id	Distinct	distinct_region_id	区域个数	个
可用区	az_id	Distinct	distinct_az_id	可用区个数	个

路径属性

指标量	请求字段	算子	响应字段	统计值	单位
客户端 VPC	l3_epc_id_0	Distinct	distinct_l3_epc_id_0	客户端 VPC 个数	个

客户端子网	subnet_id_0	Distinct	distinct_subnet_id_0	客户端子网个数	个
客户端 IP 地址	ip_0	Distinct	distinct_ip_0	客户端 IP 地址个数	个
客户端容器节点	pod_node_id_0	Distinct	distinct_pod_node_id_0	客户端容器节点个数	个
客户端命名空间	pod_ns_id_0	Distinct	distinct_pod_ns_id_0	客户端命名空间个数	个
客户端工作负载	pod_group_id_0	Distinct	distinct_pod_group_id_0	客户端工作负载个数	个
客户端 POD	pod_id_0	Distinct	distinct_pod_id_0	客户端 POD 个数	个
客户端宿主机	host_id_0	Distinct	distinct_host_id_0	客户端宿主机个数	个
客户端云服务器	vm_id_0	Distinct	distinct_vm_id_0	客户端云服务器个数	个
客户端区域	region_id_0	Distinct	distinct_region_id_0	客户端区域个数	个
客户端可用区	az_id_0	Distinct	distinct_az_id_0	客户端可用区个数	个
服务端 VPC	l3_epc_id_1	Distinct	distinct_l3_epc_id_1	服务端 VPC 个数	个
服务端子网	subnet_id_1	Distinct	distinct_subnet_id_1	服务端子网个数	个
服务端 IP 地址	ip_1	Distinct	distinct_ip_1	服务端 IP 地址个数	个
服务端容器节点	pod_node_id_1	Distinct	distinct_pod_node_id_1	服务端容器节点个数	个
服务端命名空间	pod_ns_id_1	Distinct	distinct_pod_ns_id_1	服务端命名空间个数	个
服务端工作负载	pod_group_id_1	Distinct	distinct_pod_group_id_1	服务端工作负载个数	个
服务端 POD	pod_id_1	Distinct	distinct_pod_id_1	服务端 POD 个数	个
服务端宿主机	host_id_1	Distinct	distinct_host_id_1	服务端宿主机个数	个
服务端云	vm_id_1	Distinct	distinct_vm_id_1	服务端云服	个

服务器				务器个数	
服务端区域	region_id_1	Distinct	distinct_region_id_1	服务端区域个数	个
服务端可用区	az_id_1	Distinct	distinct_az_id_1	服务端可用区个数	个

其他属性

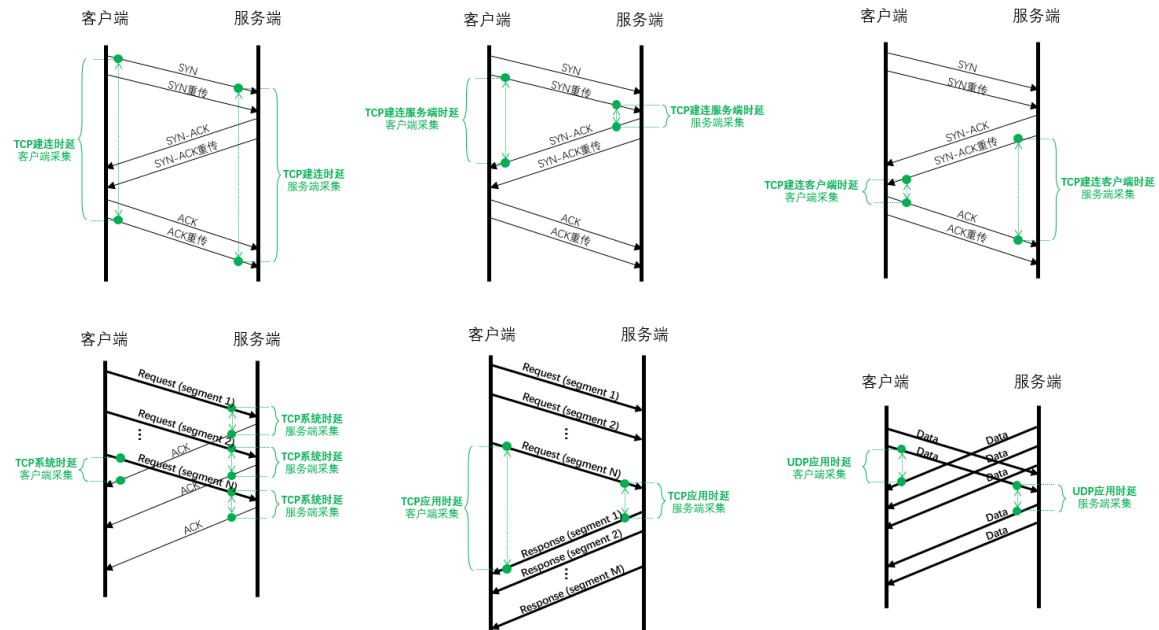
指标量	请求字段	算子	响应字段	统计值	单位
IP 类型	ip_version	Distinct	distinct_ip_version	IP 类型种数	种
采集点	tap_type	Distinct	distinct_tap_type	采集点个数	个
采集器	vtap_id	Distinct	distinct_vtap_id	采集器个数	个
网络协议	protocol	Distinct	distinct_protocol	网络协议种数	种
服务端端口	server_port	Distinct	distinct_server_port	服务端端口个数	个

资源事件

指标量	请求字段	算子	响应字段	统计值	单位
总事件	event	Sum	sum_event	总事件个数	次
创建/同步事件	sync_event	Sum	sum_sync_event	创建/同步事件个数	次
删除事件	del_event	Sum	sum_del_event	删除事件个数	次
状态改变事件	update_state_event	Sum	sum_update_state_event	状态改变事件个数	次
迁移事件	migrate_event	Sum	sum_migrate_event	迁移事件个数	次
增加 IP 事件	add_ip_event	Sum	sum_add_ip_event	增加 IP 事件个数	次
删除 IP 事件	del_ip_event	Sum	sum_del_ip_event	删除 IP 事件个数	次
重建事件	rebuild_event	Sum	sum_rebuild_event	重建事件个数	次

指标量计算方法

传输层时延



TCP 时延

关于统计周期：1s 数据源的统计周期为 1 秒，1m 数据源的统计周期为 1 分钟。

传输层时延说明：- **TCP 建连时延** - 指标意义：反映了 TCP 完成三次握手的整体时延，包括握手阶段的网络传输时延和网络传输质量，当握手阶段没有重传时约为网络传输时延的 2 倍，即等于 RTT (Round Trip Time) - 计算方法：握手阶段第一个 SYN 包和最后一个 ACK 包之间的时差。为了确保最后一个 ACK 包出现时能记录建连时延，当客户端由于重传多次发送 ACK 时，每个有 ACK 的统计周期均会记录一次建连时延 - 噪音处理：为了避免丢包、乱序包的影响，限制时延计算的最大值为 30s - **TCP 建连服务端时延** - 指标意义：在客户端观测到的该值约为网络传输时延的 2 倍，即等于 RTT - 计算方法：客户端 SYN 包与紧邻 (seq+1 和 ack 一致) 服务端 SYN-ACK 包的时间差 - 噪音处理：为了避免丢包、乱序包的影响，限制时延计算的最大值为 30s - **TCP 建连客户端时延** - 指标意义：在服务端观测到的该值约为网络传输时延的 2 倍，即等于 RTT - 计算方法：服务端 SYN-ACK 包与紧邻客户端 ACK 包的时间差 - 噪音处理：为了避免丢包、乱序包的影响，限制时延计算的最大值为 30s - **TCP 系统时延** - 指标意义：在服务端观测到的该值反映了系统 TCP 协议栈的处理时延 - 计算方法：客户端 Request 的最后一个分段报文与紧邻服务端 Payload=0 的 ACK 报文的时间差 - Request 报文的判断标准为：Payload>1、不含 SYN/FIN/RST - 注：Payload=1 的报文为 TCP Keep-Alive 报文 - 噪音处理：为了避免丢包、乱序包的影响，限制时延计算的最大值为 10s - **应用时延** - 指标意义：在服务端观测到的该值反映了应用程序的处理时延 - TCP 计算方法：客户端 Request 的最后一个分段报文与紧邻服务端 Response 的第一个分段报文的时间差 - Request、

Response 报文的判断标准为: Payload>1、不含 SYN/FIN/RST - 注: Payload=1 的报文为 TCP Keep-Alive 报文 - UDP 计算方法: 相邻客户端请求和服务端响应的时间差 - 噪音处理: 为了避免丢包、乱序包的影响, 限制时延计算的最大值为 30s

关于 TCP Keep-Alive 的额外原理说明: - 根据 keepalive 包的发送方是 TCP 连接建立的客户端还是服务端, keepalive 包的 seq 和 ack 号计算方式不同 - 发送方是客户端, keepalive 请求包的 seq 等于前一个非 keepalive 包的 seq-1, ack 等于前一个非 keepalive 包的 ack - 发送方是服务端, keepalive 请求包的 seq 等于前一个非 keepalive 包的 ack-1, ack 等于前一个非 keepalive 包的 seq - 存在 keepalive 的前一个非 keepalive 包的 seq 和 ack 相同的情况 - 此时仅根据 seq 和 ack 无法判断发送方是客户端还是服务端, 但无需判断, 只是存在这样的现象 - Wireshark 中对于 keepalive 包的判断, 是利用了 keepalive 包的 seq 和 ack 值, 需同时满足三个条件: - payload 为 0 或 1 - 为非 syn、非 fin、非 rst 等包 - seq = -1 - 注: Wireshark 在收到每个包的时候会预测和当前包同方向的下一个包的 seq, 并作为字段保存下来

平均 XX 时延和最大 XX 时延的计算说明: - 平均时延为在统计周期内监控到的所有时延的平均值 - 最大时延为在统计周期内监控到的所有时延的最大值 - 平均时延、最大时延均可继续通过算子计算出平均时延峰值、最大时延谷值等

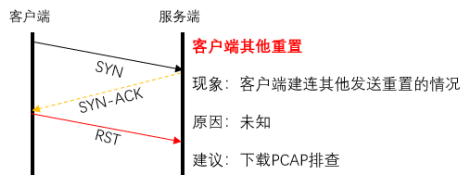
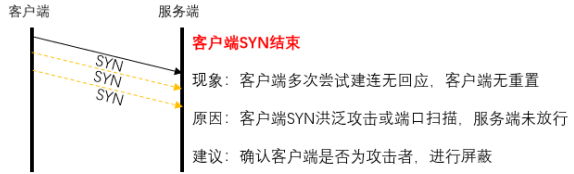
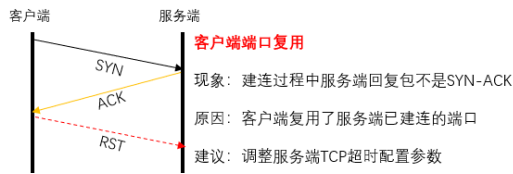
传输层性能

- 重传
 - 建连阶段: 同一个方向的两个包 Seq 一样, 即认为是重传
 - 传输阶段: 同一个方向的两个包 Seq 和 Payload 长度 (仅考虑 Payload>1 的情况) 一样, 即认为是重传
- 零窗
 - 统计 TCP 头传输窗口 size 为 0 的包

传输层异常

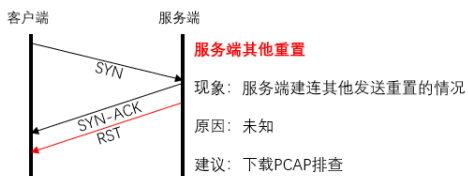
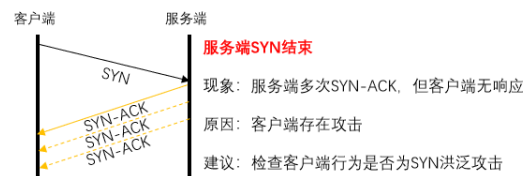
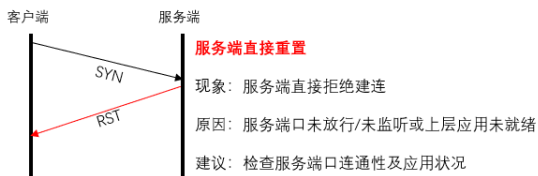
根据网流聚合状态机统计重置、半开、半关的连接数量。

TCP 客户端建连异常:



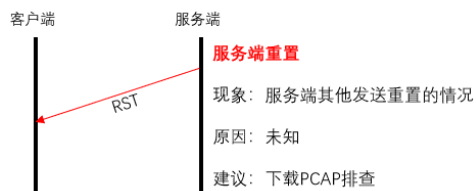
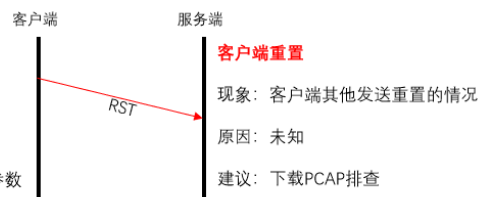
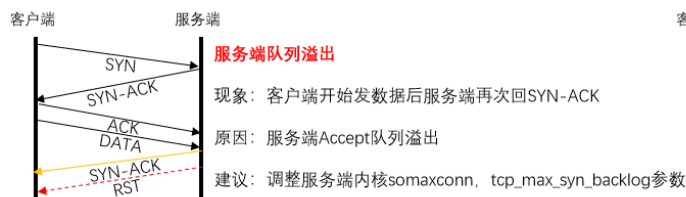
TCP 客户端建连异常

TCP 服务端建连异常：



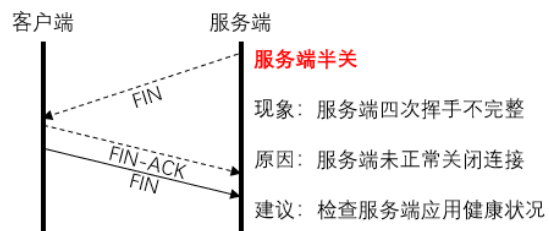
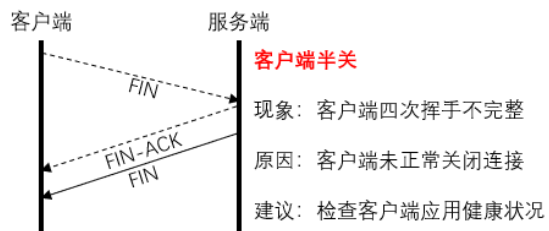
TCP 服务端建连异常

TCP 传输异常：



TCP 传输异常

TCP 断连异常：



TCP连接超时

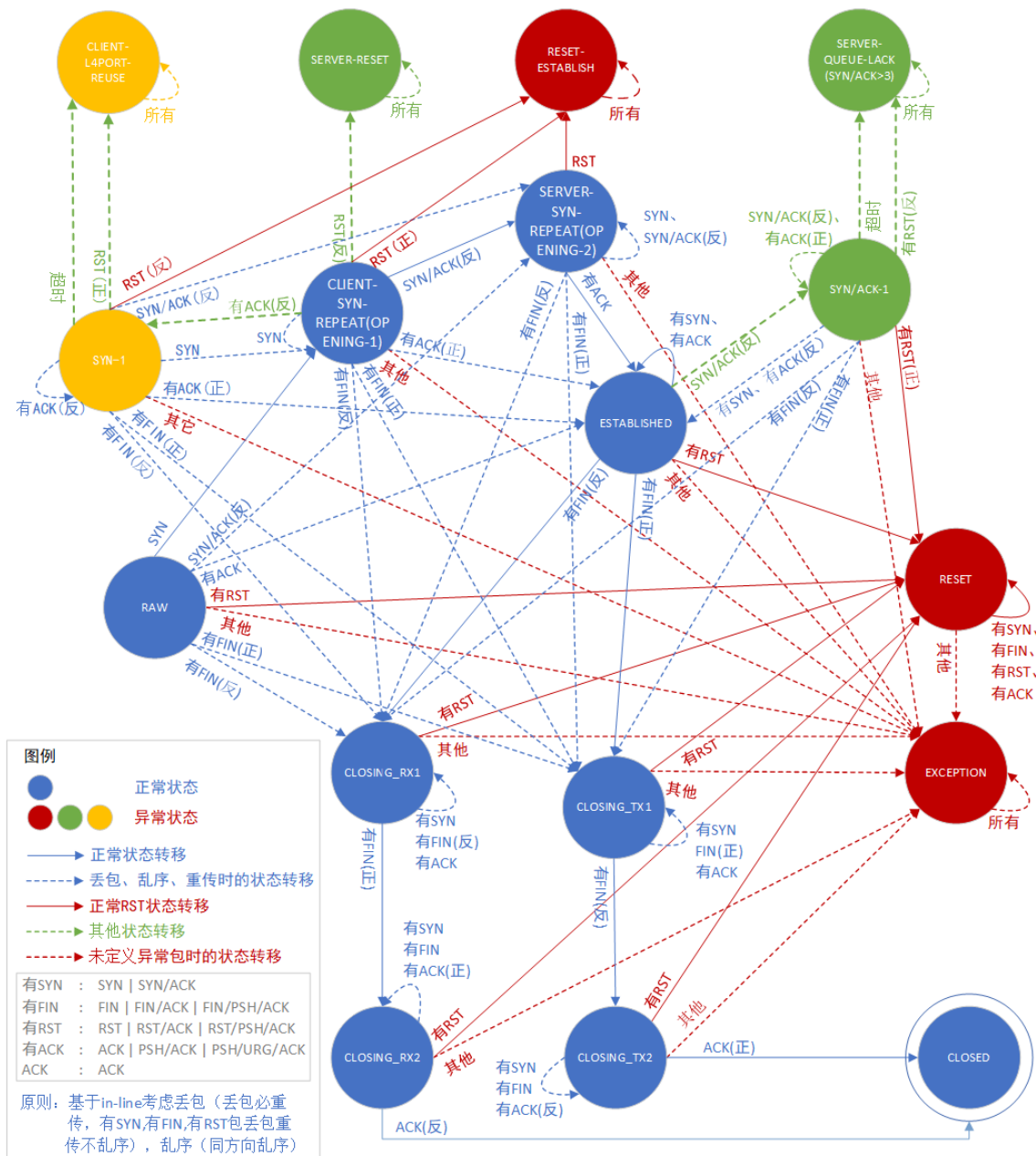
现象：连接未关闭，但长时间无数据传输

原因：主机离线或应用异常

建议：检查主机或应用健康状况

TCP 断连异常

流聚合状态机:



流聚合状态机

- 默认超时时间 (可通过 trident.yamll 进行配置)
 - RAW、OPENING_1、OPENING_2、RESET、EXCEPTION、SYN-1、CLIENT-L4PORT-REUSE、SERVER-RESET、SERVER-QUEUE-LACK、RESET-ESTABLISH 状态网流超时时间为 5 秒
 - TCP 流量 ESTABLISHED、SYN/ACK-1 状态超时时间为 300 秒
 - CLOSING_RX1、CLOSING_RX2、CLOSING_TX1、CLOSING_TX2 状态超时时间为 35 秒

- 其他协议流量仅单方向有报文时超时时间为 5 秒，双方向有报文时超时时间为 35 秒

应用层时延

- HTTP 时延
 - HTTP 请求首包和 HTTP 响应首包的时差
 - 为了避免丢包、乱序包的影响，限制时延计算的最大值为 1800s
- DNS 时延
 - DNS 请求首包和 DNS 响应首包的时差
 - 为了避免丢包、乱序包的影响，限制时延计算的最大值为 150s（参考 1-Linux 最大超时可达 150s、参考 2-Windows 最大超时一般为 10s）
- 平均 XX 时延和最大 XX 时延的计算方法与传输层时延相同

应用层异常

- HTTP 异常：参考[维基百科 List of HTTP status codes](#)，包含以下两种
 - HTTP 客户端异常：HTTP 状态码 4XX
 - HTTP 服务端异常：HTTP 状态码 5XX
- HTTP 服务端超时：TCP 流结束时还未采集到服务端响应包。另外，超过内存缓冲区大小（trident.yaml 中的 flow-count-limit，默认 100 万）的请求会按先进先出的顺序被标记为超时。
- DNS 异常：参考[RFC 2929 Section 2.3](#)，包含以下两种
 - DNS 客户端异常：DNS 响应码为 0x1，0x3
 - DNS 服务端异常：DNS 响应码非 0x0、0x1、0x3 均计算为服务端异常
- DNS 服务端超时：TCP/UDP 流结束时还未采集到服务端响应包。另外，超过内存缓冲区大小（trident.yaml 中的 flow-count-limit，默认 100 万）的请求会按先进先出的顺序被标记为超时。

传输层负载

活跃连接计算逻辑： - 采集器以四元组（客户端 IP、服务端 IP、协议、服务端口）为单位统计原始活跃连接数，并进而计算出资源、路径对应的活跃连接数 - 数据源对应的时间间隔内能采集到流量，则统计活跃连接，但存在一些特殊情况： - 1s 数据源：描述每秒统计到的活跃连接数 - 每分钟第 1 秒：包括该秒内无流量但未结束的连接，一般可用于评估并发连接（无交叠且持续时间小于一秒的多个连接会带来一些误差） - 每分钟后 59 秒：如果相同四元组的多条流在该秒内均无流量，这一秒会忽略该四元组对应的连接数，一般可用于评估并发连接的下界 - 1m 数据源：描述每分钟统计到的活跃连接数 - 包括没有流量但仍未结束的连接，一般可用于评估并发连接的上界 - 自定义数据源：根据 1s/1m 数据源通过 Avg/Max/Min 计算得到，含义与直接使用 1s/1m 数据源并选择 Avg/Max/Min 算子得到的值相同

流方向判断

采集器会将相同五元组的流量聚合到一条流（Flow），一般会认为一条流中第一个包的方向为流的方向，该方向上的源端为客户端、目的端为服务端。

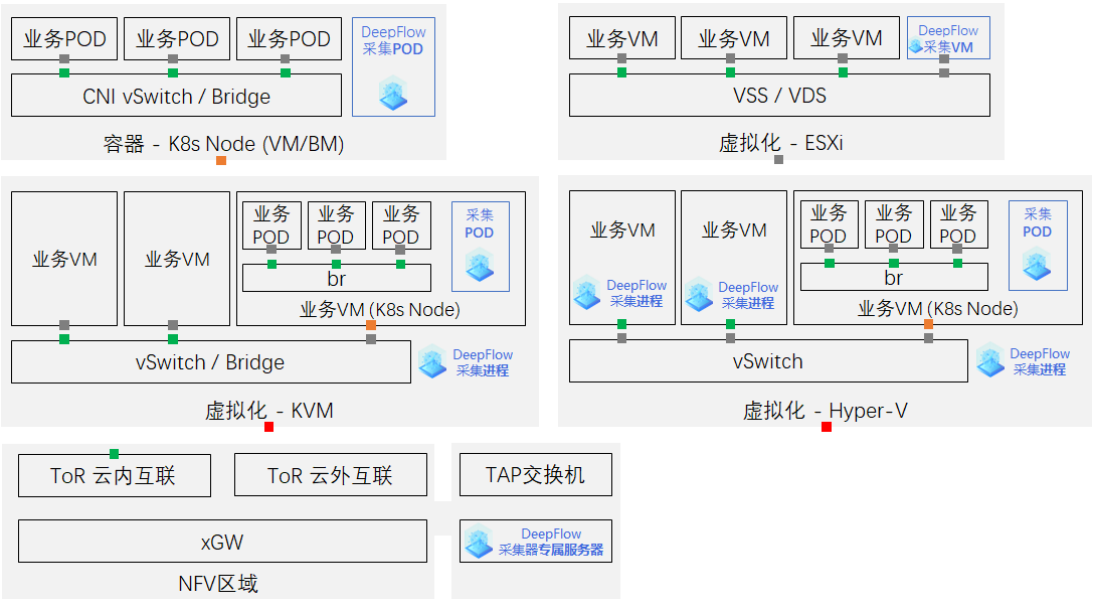
除了根据首包方向判断流的方向以外，采集器还会进行一些矫正： - 历史上曾经发送过 SYN 包的一端会被认为是客户端 - 历史上曾经发送过 SYN+ACK 包的一端会被认为是服务端 - 若无法通过上述特殊 TCP 标志位进行判断，则会继续如下矫正： - 历史上作为首包目的端出现得更多的一端会被认为是服务端 - 如果两侧历史上作为首包目的端出现的次数非常相近时，如果一侧端口号小于 32768 另一侧大于等于 32768，则认为小端口号的一端是服务端（[Many Linux kernels use the port range 32768-60999](#)）

虚拟网络链路追踪

基于对虚拟网络的深入理解，全景图可展现资源之间通信路径的全链路追踪数据。

建议： 注意：广播和组播流量仅会在客户端位置记录统计值（对于有应答的 ARP 流量也会在服务端位置记录），其他网卡上的广播/组播流量都会标记为统计位置=其他。

客户端/服务端侧



客户端/服务端侧链路追踪

容器环境

容器环境下采集器对 POD 虚接口和容器节点出接口进行采集： - POD 虚接口：上图绿色接口 - 该类接口的流量，其统计位置属性将会被标记为客户端或服务端 - 容器节点出接口：上图橙色接口 - POD 通过隧道封装、路由转发进出容器节点的流量，

流量最外层 MAC 为出接口 MAC，其统计位置属性将会被标记为客户端容器节点或服务端容器节点 - POD 通过 SNAT、DNAT 进出容器节点的流量，流量 MAC 为出接口 MAC，其统计位置属性将会被标记为客户端或服务端 - POD 通过 vSwitch 交换转发进出容器节点的流量，**会被采集器标记为统计位置=其他** - 容器节点（包括使用 HostNetwork 命名空间网卡的 POD）直接产生的流量，流量 MAC 为出接口 MAC、IP 为出接口 IP，其统计位置属性将会被标记为客户端或服务端

ESXi 环境

ESXi 环境下采集器对业务虚拟机的虚接口（上图绿色接口）进行采集，其统计位置属性将会被标记为客户端或服务端。

注意：当业务虚拟机中运行容器且安装了容器类型采集器时，请勿将该虚拟机的流量镜像至 ESXi 类型采集器。

KVM 环境

KVM 环境下采集器对虚拟机虚接口和宿主机出接口进行采集：- 虚拟机虚接口：上图业务 VM 的绿色接口 - 该类接口的流量，其统计位置属性将会被标记为客户端或服务端 - 当虚拟机运行容器，且 DeepFlow 同步了容器集群信息时，KVM 采集器会忽略该类虚拟机虚接口的流量，需要通过容器类型的采集器在内部采集 - 宿主机出接口：上图红色接口 - 虚拟机（包括其上 POD）通过隧道封装进出宿主机的流量，流量最外层 MAC 为出接口 MAC，其统计位置属性将会被标记为客户端宿主机或服务端宿主机 - 虚拟机（包括其上 POD）通过 vSwitch 交换转发进出宿主机的流量，其统计位置属性将会被标记为客户端宿主机或服务端宿主机 - 注意：如果采集器的采集网口中，除了虚拟机的虚接口以外会多次采集到同一份流量，此时会导致对应统计位置上的统计值翻倍

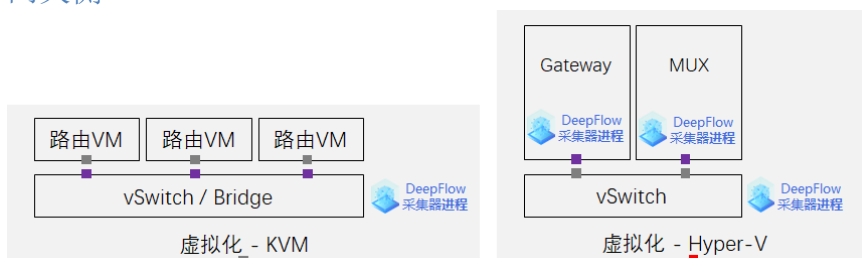
Hyper-V 环境

Hyper-V 环境下采集器对宿主机出接口进行采集，其上业务虚拟机内部也许部署采集器：- 业务虚拟机内部：上图业务 VM 的绿色接口 - 该类接口的流量，其统计位置属性将会被标记为客户端或服务端 - 对于运行容器的业务虚拟机，统计位置属性的标记请参考**容器环境**章节的描述 - 宿主机出接口：上图红色接口 - 虚拟机通过 vSwitch 交换转发进出宿主机的流量，其统计位置属性将会被标记为客户端宿主机或服务端宿主机，**注意也包括宿主机直接产生的流量** - 虚拟机通过隧道封装、SNAT、DNAT 进出宿主机的流量，其统计位置属性将会被标记为客户端宿主机或服务端宿主机

NFV 接入交换机镜像

在腾讯 TCE 场景下，NFV 区域的流量可通过接入交换机镜像至专属服务器采集器，对云内互联 ToR 上联口的镜像流量可选择将其采集点标记为虚拟网络，此时该流量的统计位置属性将会被标记为客户端或服务端。

网关侧



网关侧链路追踪

KVM 环境

KVM 环境下若所有虚拟机均为做路由的虚拟机时，采集器对虚拟机虚接口（上图紫色接口）进行采集，其统计位置属性将会被标记为客户端网关或服务端网关。

注意：对于宿主机上虚拟机作为路由器的场景，请勿采集出接口流量，避免混淆。

Hyper-V 环境

Hyper-V 环境下若其上虚拟机作为 BGP Gateway 或 MUX 时，宿主机上采集器对出接口进行采集，其上虚拟机内部也建议部署采集器：- 虚拟机内部：上图 Gateway 和 MUX 的紫色接口 - 若为虚拟机源发流量，其统计位置属性将会被标记为客户端或服务端 - 若为虚拟机 NAT 或 LB 流量，其统计位置属性将会被标记为客户端网关或服务端网关 - 宿主机出接口：上图红色接口 - 其统计位置属性将会被标记为客户端网关宿主机或服务端网关宿主机

流量日志写入限速

默认每个数据节点每秒支持分别写入 50000 条一般流日志, HTTP 流日志和 DNS 流日志，如此值过大或过小，可通过如下方式修改：

```
##### 修改 /usr/local/deepflow/saltstack/pillar/droplet.sls  
throttle: 50000
```

修改完毕后，执行 `salt -N tsdb state.apply` 使修改生效。

流量下载

功能使用

流量下载

通过点击全景图的流量下载，可创建、查看以及修改 PCAP 策略，PCAP 策略能针对虚拟网络以及接入网络的计算资源（云服务器）、网络资源（IP 地址、子网）、容器资源（工作负载、容器服务）、业务画像（资源组）等维度生成的 PCAP 文件，也能进行采集器、ACL 策略的过滤，让用户更关注于所需报文：- PCAP 策略的后端匹配规则与分发策略的后端匹配规则一样，可参考分发功能 - 可根据流量来源来

确定需要进行 PCAP 文件保存的对象是虚拟网络还是接入网络的镜像口 - PCAP 策略的 Payload 截断，可设置，默认为 0，即表示截断为 0，只截取包头 - 采集点选择虚拟网络时，截断长度支持 0-65535 的整数，当输入值大于 1000 时，需将采集器 PCAP 套接字配置为 TCP，否则实际生效值仅为 1000 - 采集点选择物理网络时，截断长度支持 0-65535 的整数 - 名称点击时可查看 PCAP 策略生成的文件的个数和大小曲线详情图 - PCAP 策略生成的文件是按照后端配置文件设置的最大超时时间与最大切割文件大小共同生效的，默认超时时间为 5min，切割文件大小为 25M - PCAP 文件个数：此策略当前系统中存在的 PCAP 文件的个数，包含最新的用户不可见的 tmp 文件 - PCAP 文件大小：此策略当前系统中存在的 PCAP 文件大小的总和，包含最新的用户不可见的 tmp 文件

通过页面点击 PCAP 策略的查看详情按钮，可选择任意时间范围内的生成的 PCAP 文件，并对需要下载的 PCAP 文件进行选择，然后点击直接下载或者间接下载： - 采集器名称：对应生成 PCAP 文件的采集器名称 - 采集网口识别：对应采集器的采集网口的 MAC 地址 - 直接下载：直接将选择的 PCAP 文件进行合并下载 - 过滤下载：用户可对选中的文件按 IP 类型，协议，端口以及 IP 进行筛选下载。 - 过滤下载端口：是指 PCAP 文件的匹配的任意源端口或者目的端口 - 过滤下载 IP：是指 PCAP 文件的匹配的任意源 IP 或者目的 IP

典型场景

辅助全景图诊断

在全景图流量搜索中完成故障诊断之后，可在此下载详细的 PCAP 包。注意需要事先配置合适的 PCAP 存储策略。

若故障诊断未能完全定位，用户也可预先在此处设置 PCAP 存储策略，以辅助下次出现故障时的诊断。

功能规格

规格

无。

授权

PCAP 功能需要获取 DeepFlow 监控诊断授权，请向云杉网络服务人员申请。

限制

- 页面显示的流量统计数据以匹配 PCAP 策略的流量为准，异常情况下可能由于系统负载过高未正常存储 PCAP，这类异常情况下用户可通过系统告警知晓
- PCAP 策略中 payload 等于 0 时，PCAP 文件的包头非全包头，如下字段不会传输：
 - IPV4: Type of Service、Checksum、Options

- IPV6: Traffic Class
- TCP Header: Checksum、Urgent Pointer
- TCP Options: WIN_SCALE、MSS、SACK 传输，其余都不传输
- UDP: Checksum

限制

无。

流量分发

通过配置分发策略，用户能精准地将整个虚拟网络以及物理网络中指定流量以最高效的方式分发到多个流量分析工具池（分发点）。

功能使用

特性介绍

面向业务的流量过滤

用户可设置源端地址、端口号、目的端地址、端口号、协议五元组作为过滤条件，并支持直接输入 VPC、子网、云服务器、容器 POD、自定义资源组等替代 IP 地址，精准过滤虚拟网络中的流量，最大利用网络带宽和分析工具效率。

源端 Payload 截断

用户可设置对网包的 L4 Payload 长度进行截断，最大利用网络带宽和分析工具效率。

多层流量标签

分发流量利用隧道封装发送，通过在隧道头中携带特殊的流量标签，用于对不同 VPC、容器服务、流量采集位置（客户端/服务端）等流量属性进行标记，以帮助后端分析工具解决 VPC 之间 IP 段冲突、容器服务后端 POD 的 IP 地址频繁变动等问题，并实现对客户端、服务端采集流量进行端到端对比分析。

采集器支持在原始包之前添加 VXLAN 或 ERSPAN 隧道及内层 VLAN 标签，后端汇聚分流设备或 TAP 交换机仅需要做一次性的隧道解封装配置，无需根据不同的隧道标签解封装报文的 VLAN 标签，极大简化分流设备/TAP 交换机的配置复杂度和特性依赖。

流量全局去重

对于同一个网包，采集器会在源端云服务器及目的端云服务器处多次采集到。当发往同一个分发点时，如果不做去重，后端分析工具将收到重复的网包，同时也会占用至少双倍的带宽。DeepFlow 引入分布式去重功能解决重复采集的问题，具体原理可查看场景介绍。

为了最大限度减少对带宽的占用，对发往同一个分发点、设置相同隧道标签的同一个网包，按所有匹配策略的最大 **Payload** 截取长度进行截取。例如两条分发策略 **A=Payload 截断 30**、**B=Payload 截断 20**，对于同时匹配这两条策略的网包将会被截取 30 字节 **Payload** 后进行发送。

流量多路分发

当一个网包匹配的多条分发策略对应不同分发点或不同隧道标签时，流量将会复制多份分别发送。例如两条策略 **A=分发点 C1**、**B=分发点 C2**，对于同时匹配这两条策略的网包，在分发时复制两份，一份分发给 **C1**，一份分发给 **C2**。

资源变更感知

当云服务器迁移、云服务器 **IP** 变更、容器 **POD** 弹性伸缩时，分发策略自动感知，无需用户重新设置，迁移期间的流量持续分发。

配置说明

分发策略

一条分发策略由采集点、采集器、采集点过滤规则、对端过滤规则、流量处理方法和转发规则六部分组成。

采集点标记流量采集的位置，分为物理采集点和虚拟采集点。采集器可对流量进行采集器过滤。采集点过滤可以是指定 **VPC** 中的一个 **IP** 地址或 **IP** 段、一台云服务器、一个子网、一个容器 **Deployment** 等资源，用户也可通过指定协议和端口号对采集点的流量进行过滤。除此之外，通过指定对端过滤规则，用户也可对采集点过滤的资源的通信对端做精准的限制，实现网络五元组的全方位过滤。

策略自动删除：分发策略关联资源被删除后，分发策略自动删除。当资源为工作负载、容器服务时，会验证其所在命名空间是否被删除，若命名空间也被删除，则分发策略自动删除

新建/修改说明：

- 采集点过滤规则：计算资源（云服务器）、网络资源（**IP** 地址、子网）、容器资源（工作负载、容器服务）、业务画像（资源组），上述资源对应的云服务器为实际发生采集分发的点，工作负载则对于实际发生采集分发的 **POD**。端口号，如果为采集器发送出去的流量则为源端口；如果为采集器接收的流量则为目的端口号。
- 对端过滤规则：从采集点角度来看，采集点之外的都称为对端。比如采集器为云服务器 **A**，需要统计云服务器 **B** 与云服务器 **A** 之间的流量，则云服务器 **B** 对于采集点云服务器 **A** 来说就是对端。对端配置端口时，则在采集点上匹配发送出去的流量的目的端口；采集器匹配接收的流量的源端口。

- **PayLoad 截断：**默认分发的是全包头+全 PayLoad。PayLoad 配置数据则分发的为全包头 + PayLoad 截断的长度。
- **流量标签：**对应为 Vxlan 的 VNI、GRE 的 KEY 和 Erspan 的 Session ID。
- **分发策略中：**云服务器显示为(名称，别名，区域)，子网显示为(网段，名称，别名)

分发点

分发点是接收分发流量的隧道端点，可以是交换机上接收隧道的接口 IP，也可以是具备解析隧道协议的分析工具所在的宿主机/云服务器的接口 IP。

新建/修改说明：

- **隧道类型：**目前支持 VXLAN 和 ERSPAN

删除说明：

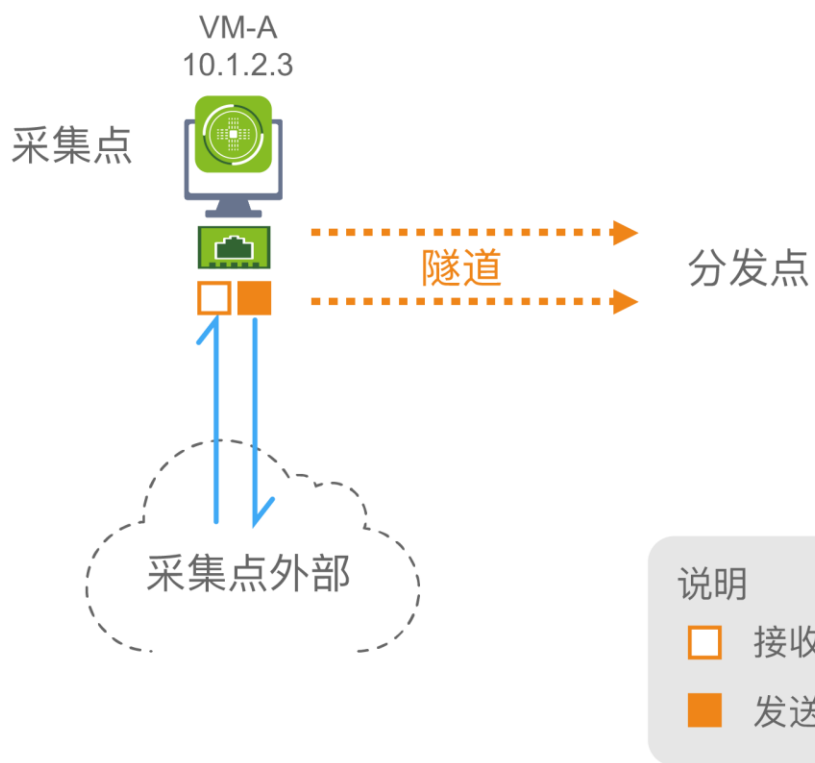
- 可删除没有分发策略引用的分发点

典型场景

采集点为单个云服务器

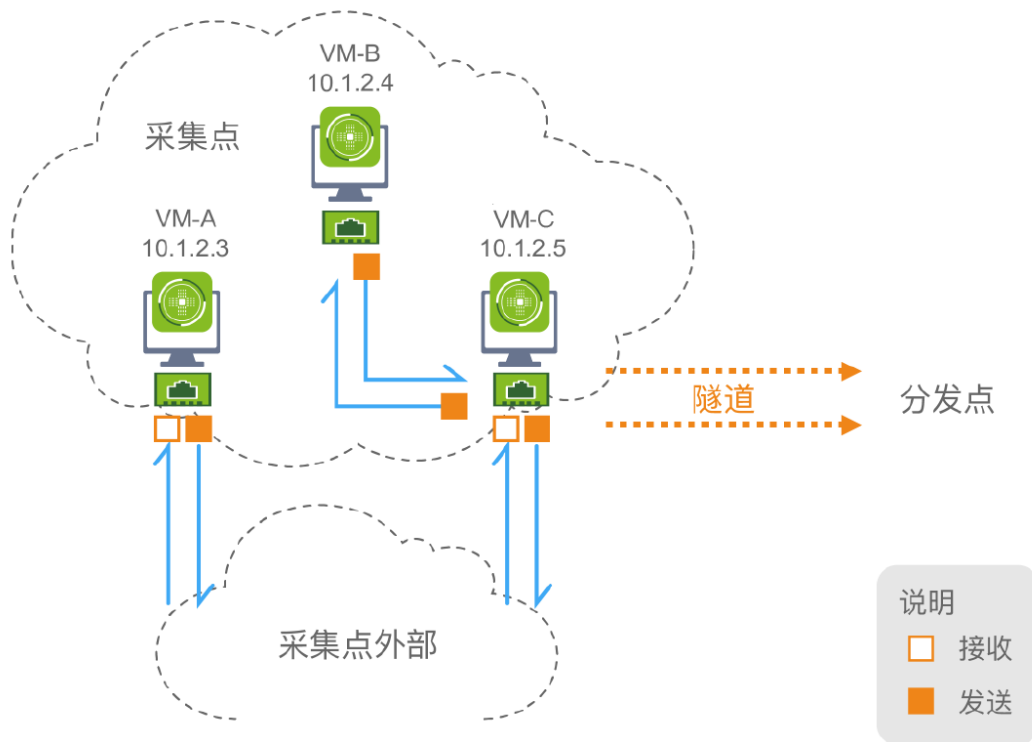
当用户选择单个云服务器或者单个 IP 地址时，具体的流量采集点为云服务器（IP 地址可映射到具体的云服务器）在宿主机上的虚拟接口。如下图所示，在 VM-A 的虚拟接口上采集到的接收和发送方向流量，经过过滤规则之后都将会被分发到指定的分发点。

建议：注意：对于使用了安全组的 OpenStack 环境，虚拟接口上发送方向的流量是经过安全组之间的，接收方向的流量是经过安全组之后的。



采集点为 IP 段或子网

当用户选择一个 IP 段或子网时，具体的流量采集点为这些 IP 地址映射到的所有云服务器在宿主机的 Tap 接口。如下图所示，当采集点与外部的通信时，对采集点上接收和发送的流量进行分发，这与采集点为单云服务器场景相同。当采集点内部通信时，为达到全局去重效果，只会将流量发送端的虚拟接口上做单向采集分发。例如下图中 VM-B 与 VM-C 之间的通信，只会在 VM-B 和 VM-C 上采集分发各自发送方向上的流量。



流量标签的应用

容器环境中 POD 会频繁扩缩容，导致 POD 的 IP 会频繁变化。为了让后端分析工具识别一组 POD 提供的服务，可创建工作负载粒度的分发策略，并填写合适的隧道标签。此时后端分析工具通过识别隧道标签即可区分不同的容器服务流量，无论 POD 如何增删和 IP 变更都不会对流量识别产生影响。

关闭全局去重

采集器分发流量中支持在隧道头部和内层 VLAN 中记录流量采集的位置（详见二次开发章节），因此用户可以选择关闭全局去重，使得流量分析工具可以获取源、目的位置获取的两份流量，这样有助于精确定位网络性能问题发生的位置。

在系统-采集器-配置中修改包分发功能开关-全局去重可关闭全局去重能力。

开启内层 VLAN 封装

采集器支持在原始流量中加入 VLAN 头部，并将隧道 ID 和流量采集位置记录在内层 VLAN 头部中（详见二次开发章节）。打开内层 VLAN 封装后，TAP 交换机或流量汇聚设备的配置将会极大的简化，不用关心如何处理隧道头部的流量标签。

在系统-采集器-配置中修改包分发配置参数-内层附加头为 802.1Q 可开启内层 VLAN 封装。

配置举例

分发云服务器 vm1 的全部流量

- 采集点过滤规则：云服务器=vm1
- 对端过滤规则：不配置
- 匹配如下流量：
 - vm1 上分发出方向的 src_ip = vm1
 - vm1 上分发入方向的 dst_ip = vm1

发云服务器 VM1 的 tcp 协议的 80 端口的流量

- 采集点过滤规则：云服务器=vm1，协议=tcp，端口=80
- 对端过滤规则：不配置
- 匹配如下流量：
 - vm1 上分发出方向的 src_ip = vm1 and protocol = tcp and src_port = 80
 - vm1 上分发入方向的 dst_ip = vm1 and protocol = tcp and dst_port = 80

分发云服务器 vm1 与云服务器 vm2 之间的流量

- 采集点过滤规则：云服务器=vm1
- 对端过滤规则：云服务器=vm2
- 匹配如下流量：
 - vm1 上分发出方向的 src_ip = vm1 and dst_ip = vm2
 - vm1 上分发入方向的 dst_ip = vm1 and src_ip = vm2

分发云服务器 vm1 与云服务器 vm2 的 tcp 协议的 80 端口通信的流量

- 采集点过滤规则：云服务器=vm1
- 对端过滤规则：云服务器=vm2，协议=tcp，端口=80
- 匹配如下流量：
 - vm1 上分发出方向的 src_ip = vm1 and protocol = tcp and dst_ip = vm2 and dst_port = 80
 - vm1 上分发入方向的 dst_ip = vm1 and protocol = tcp and src_ip = vm2 and src_port = 80

分发 IP 段 192.168.10.24-192.168.10.26 的 80 端口流量

设定 192.168.10.24-192.168.10.30 对应的云服务器为 vm1，vm2，vm3。

- 采集点过滤规则：IP=192.168.10.24-192.168.10.26，端口：80
- 对端过滤规则：不配置
- 匹配如下流量：
 - 分发 vm1 与 vm2、vm3 产生的流量规则如下：

- vm1 分发出方向的 src_ip = vm1 and src_port = 80 and dst_ip = vm2/vm3
- 分发 vm1 与 vm2、vm3 以外的流量规则如下：
 - vm1 分发出方向的 src_ip = vm1 and src_port = 80
 - vm1 分发入方向的 dst_ip = vm1 and dst_port = 80

分发子网 192.168.10.0/10 的流量

设定 vm1 为子网 192.168.10.0/10 中的一个云服务器。

- 采集点过滤规则：CIDR=192.168.10.0/10
- 对端过滤规则：不配置
- 匹配如下流量：
 - 分发 vm1 与子网内的云服务器规则如下：
 - vm1 分发出方向的 src_ip = vm1 and dst_ip = 子网内云服务器的 IP
 - 分发 vm1 与子网内云服务器以外的流量规则如下：
 - vm1 分发出方向的 src_ip = vm1
 - vm1 分发入方向的 dst_ip = vm1

分发子网 192.168.10.0/10 与 10.0.0.0/16 子网的 udp 流量

设定 vm1 为子网 192.168.10.0/10 中的一个云服务器。

- 采集点过滤规则：CIDR=192.168.10.0/10
- 对端过滤规则：CIDR=10.0.0.0/16，协议=udp
- 匹配如下流量：
 - vm1 出方向的 src_ip = vm1 and dst_ip = 10.0.0.0/16 子网的 IP and protocol = udp
 - vm1 入方向的 dst_ip = vm1 and src_ip = 10.0.0.0/16 子网的 IP and protocol = udp

分发 vpc1 的全部流量

设定 vm1 为 vpc1 中的一个云服务器。

- 采集点过滤规则：VPC=vpc1，CIDR=0.0.0.0/0
- 对端过滤规则：不配置
- 匹配如下流量：
 - 分发 vm1 与 vpc1 内的云服务器规则如下：
 - vm1 分发出方向的 src_ip = vm1 and dst_ip = vpc1 内的云服务器 IP
 - 分发 vm1 与 vpc1 以外的流量规则如下：

- vm1 分发出方向的 src_ip = vm1
- vm1 分发入方向的 dst_ip = vm1

功能规格

规格

资源	默认限制	修改限制
分发策略数量	1000 个	无法调整
分发点数量	8 个	无法调整
资源组数量	1000 个	无法调整
资源组中云服务器数量	64 个	无法调整

授权

流量分发功能需要获取 DeepFlow 采集分发授权，请向云杉网络服务人员申请。

限制

- 页面显示的流量统计数据以匹配分发策略和分发点的流量为准，异常情况下可能由于限速、熔断的原因未成功发送流量，这类异常情况下用户可通过系统告警知晓。
- 在分发策略选择了容器服务、且容器服务同时使用 TCP 和 UDP 协议的情况下，分发策略将匹配每个服务端口号的 TCP 和 UDP 协议。这种情况下，如果服务使用的 TCP 和 UDP 端口号不相同，可能导致多匹配一部分流量。

规范

参考二次开发

网络拓扑

全景图网络拓扑功能聚焦于从逻辑、虚拟、物理的视角展现网络的配置和状态指标数据。

功能使用

逻辑拓扑

租户视角

通过指定输入需要查询的信息，可从 VPC，子网，安全组，云服务器，IP 的角度查看的资源层级以及连通性关系，通过悬浮、展开收起等功能可查看流量大小、云平台以及资源统计量等信息。通过添加查询条件到常用，可保持为常用按钮，更快点击查看。

说明： - 默认显示：如果添加常用按钮时，按照常用的第一个显示；如果未添加常用按钮，则不显示页面。 - 云服务器节点：云服务器属于多个子网时，会在每个子网都复制一个云服务器节点 - 路由节点：只支持 **openstack**，公有云，可支持查看路由表 - 安全组节点：支持 **openstack**，公有云，可悬浮查看安全组规则。每个云服务器如果关联安全组都复制一个安全组节点，没有挂任何云服务器的安全组不会出现在拓扑界面。 - 悬浮：查看节点资源管理信息和流量大小，蓝色链接点跳转到资源管理和流量统计

管理员视角

通过指定类型为宿主机，宿主机下拉框可查看具体宿主机的资源层级以及连通性关系，通过悬浮、展开收起等功能可查看流量大小、云平台以及资源统计量等信息。通过添加常用宿主机，可快速查看和点击经常使用的宿主机的资源拓扑。

说明：根据当前宿主机上的云服务器所属的 VPC 统计，宿主机的拓扑也是根据云服务器所属的 VPC，路由器，子网，安全组等关系展开。

连通性诊断

通过输入 IP，协议加端口这些查询条件，可查看 IP 的逻辑网络关系以及虚拟网络组网情况，帮助用户从逻辑网络配置层面检查连通性： - 对云服务器状态、安全组规则、对等连接和路由器表项进行判断并检测是否影响网络连通性 - 支持两端都为内网 IP 或者外网 IP 进行检测 - 安全组检测：检测时会根据安全组的优先级，以及安全组规则的优先级来进行匹配，如果安全组的允许规则被匹配则认为检测通过；如果安全组的拒绝规则被匹配或者未匹配任何安全组规则则认为检测不通过。 - 路由器表项检测：只在不同 VPC 的两个云服务器通信检测，如果 VPC 直接建立了连接则判断路由规则是否放通连接。

功能规格

规格

- 租户网络
 - 每子网云服务器展示个数：自动显示逻辑如下：
 - VPC 总云服务器个数 < 50: ALL
 - 100>= VPC 总云服务器个数 >=50: 50
 - 300>= VPC 总云服务器个数 >100: 20
 - 500>= VPC 总云服务器个数 >300: 10
 - VPC 总云服务器个数 >500: 5
 - 输入查询条件查询时
 - 界面所有节点数加起来 <= 100 个时，默认显示为节点层级全打开
 - 界面所有节点 > 100 个时，只全打开第一个节点的全部节点
 - 常用按钮：最多可创建 10 个

授权

全景图功能需要获取 DeepFlow 监控诊断授权，请向云杉网络服务人员申请。

限制

无。

规范

无。

视图

视图是用户自定义监控图表的窗口。目前用户可将全景图-流量搜索中的任意图表加入视图中，并对其中的任意折线图设置告警策略，以及对整个视图设置报表生成策略。同时可以将视图自由分享给不同账户。

功能使用

功能介绍

视图页面提供用户自定义功能可视化的能力，可将 DF 中各功能图表自由组合，并提供告警(针对时间维度呈现图表)和报表能力。

视图由子视图组成，每个子视图是一个独立的可视化功能模块，可以通过子视图的布局调整工具来灵活调整子视图的大小和位置。

视图可以加入全景图中的所有图表。

视图中每个子视图(图表)提供了一组标准化的图表操作，包含功能页中的所有操作。此外，子视图独有的操作包括：

- 告警
 - 时间维度折线图可打开告警配置界面。在该界面中，可以
 - 管理告警策略
 - 开启禁用告警策略
 - 开启禁用折线图中加载告警策略的告警事件段
 - 查看告警策略对应的告警事件
- 打开原功能页面
 - 子视图会记录加入视图时的查询过滤条件，可以通过打开原功能页面按钮返回对应的功能页面，并加载加入时的查询条件。

视图层面，支持功能包括

- 创建报表策略

对当前视图创建报表策略，定时生成可下载的离线报表。

- 全屏

将视图的内容全屏展示，适合在大屏场景下演示。

- 显示模式切换

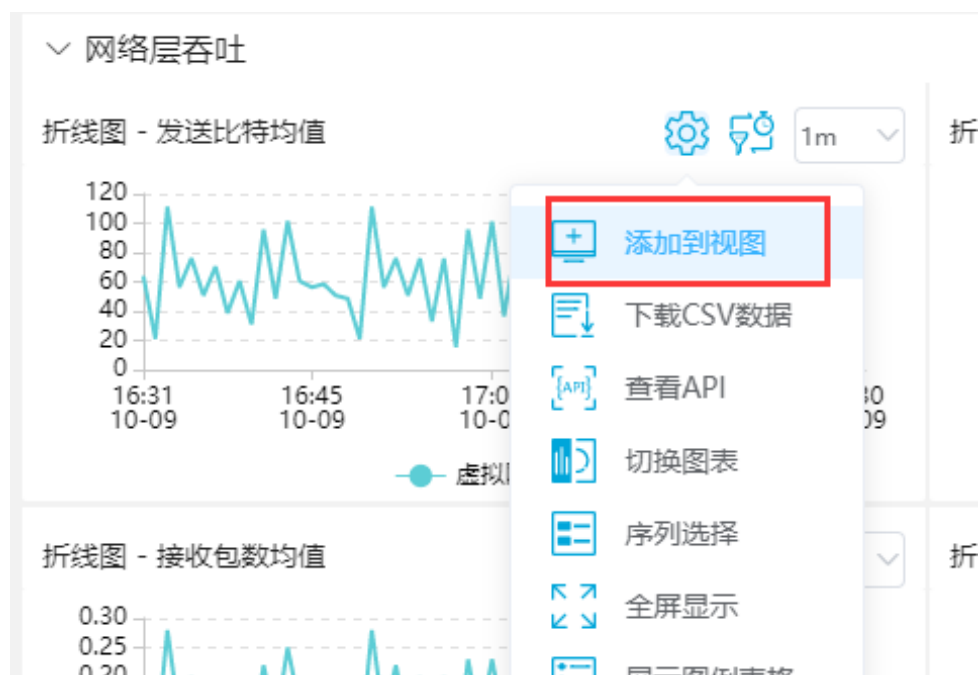
可切换至深色模式，满足大屏演示需要。

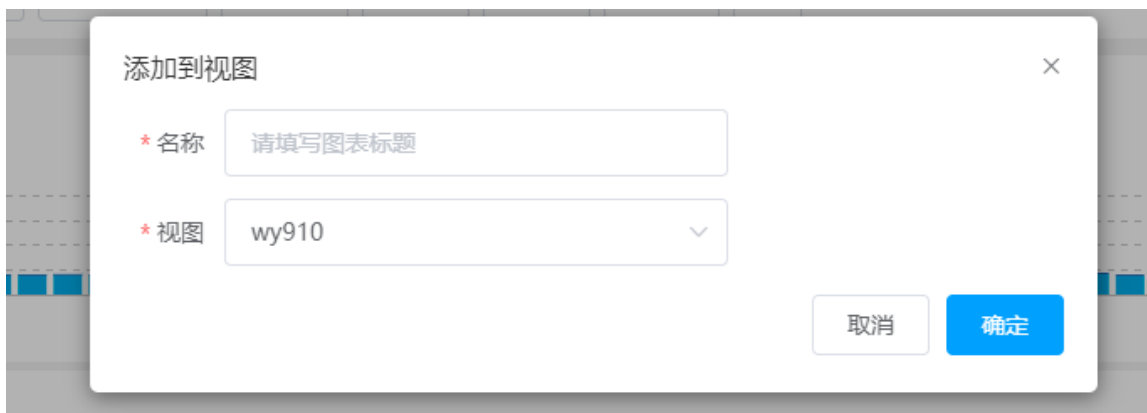
- Tip 同步

可切换趋势类子视图所有 Tip 同步显示或只显示鼠标所指子视图

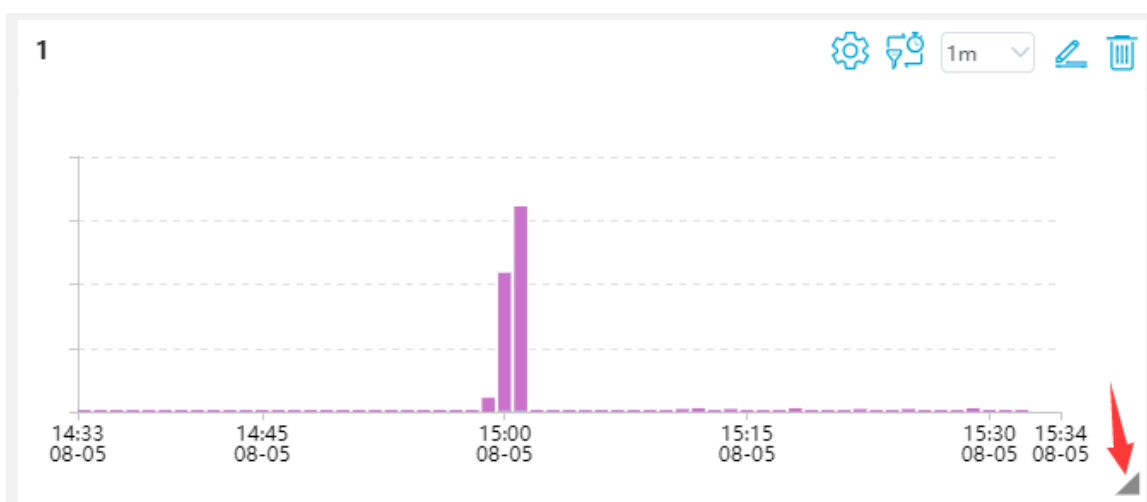
功能使用

- 全景图功能页中选择任意子视图，点击添加到视图按钮并保存

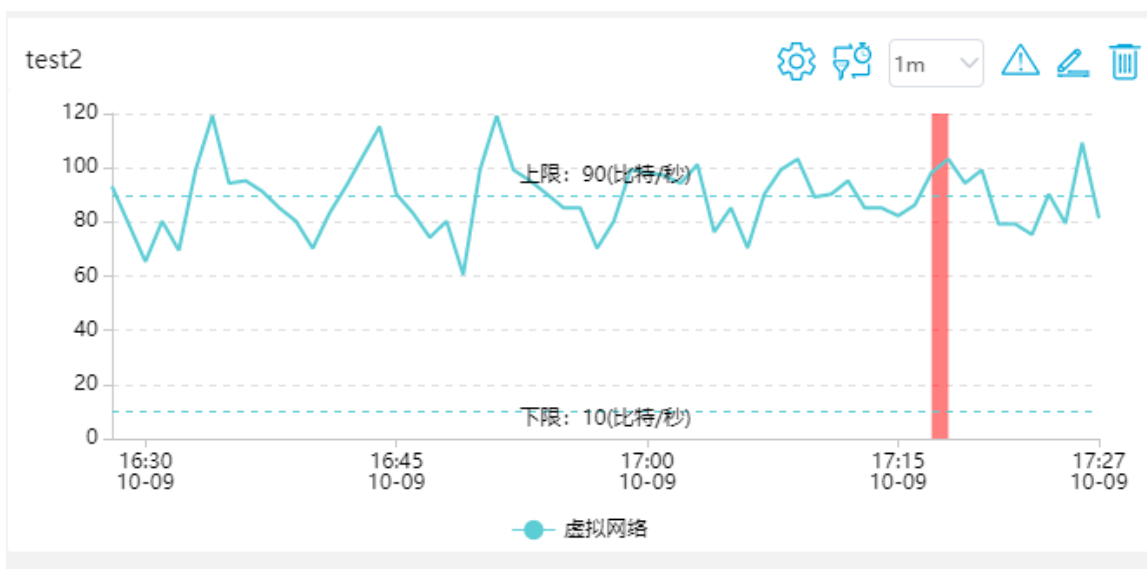




- 通过子视图的右下角调整大小，也可在标题处拖动子视图，调整视图的布局。



- 为流量设置告警策略



修改告警策略

×

* 策略名称

test

监听对象

子视图: test2

* 告警等级

☒ 低
☐ 中
☐ 高

* 数据源类型

1m

* 告警目标

虚拟网络

* 触发条件

接收比特均值

?

主指标阈值

下限

10

上限

90

标签

使用Enter键或者逗号分割标签

* 通知邮箱

a@a.com

?

取消

确定

- 对创建成功的告警策略可以启用禁用，或者进行修改、删除

告警策略

×

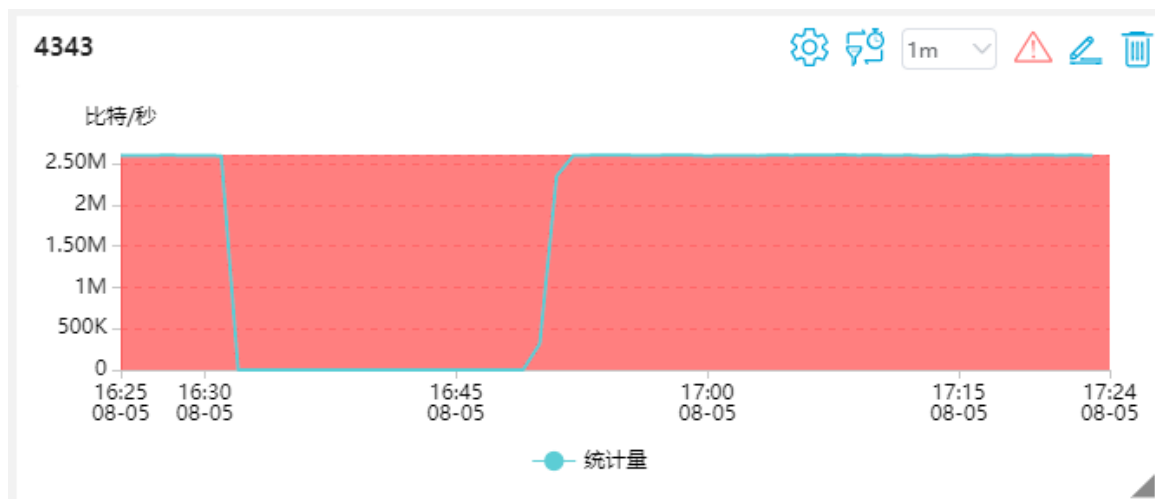
新建告警策略

策略名称	数据源	启用	展示告警	最近事件时间点	操作
1232	1m	☒	☒	现在	告警事件 修改 删除

取消

确定

- 默认状态下，折线图会自动加载对应的告警事件时间段，以红色背景显示



- 可以在视图右上角为视图创建报表策略



- 可在视图右上角分享该视图分享给单个账号、管理员账号集合、所有账号集合（单个管理员赋权优先于全部管理员/所有账号赋权），赋予其只读/读写权限，分享来的视图在页面有橘色只读/读写标识

视图分享

* 选择账号 必填

* 分配权限

只读
读写

确定

注意：当租户与管理员间有共享的视图时，管理员需要将创建此视图的租户账号锁定，才可修改此视图

- 视图支持导出为模板文件，并可导入对应的模板文件生成新的视图
- 视图支持添加模板变量，被每一个子视图引用，通过模板变量切换控制子视图的搜索条件
- 支持子视图划分模块管理，按照用户需求归类查看子视图

典型场景

构建一个用于大屏的流量监控视图

- 视图列表中创建一个新的视图，名称为流量监控

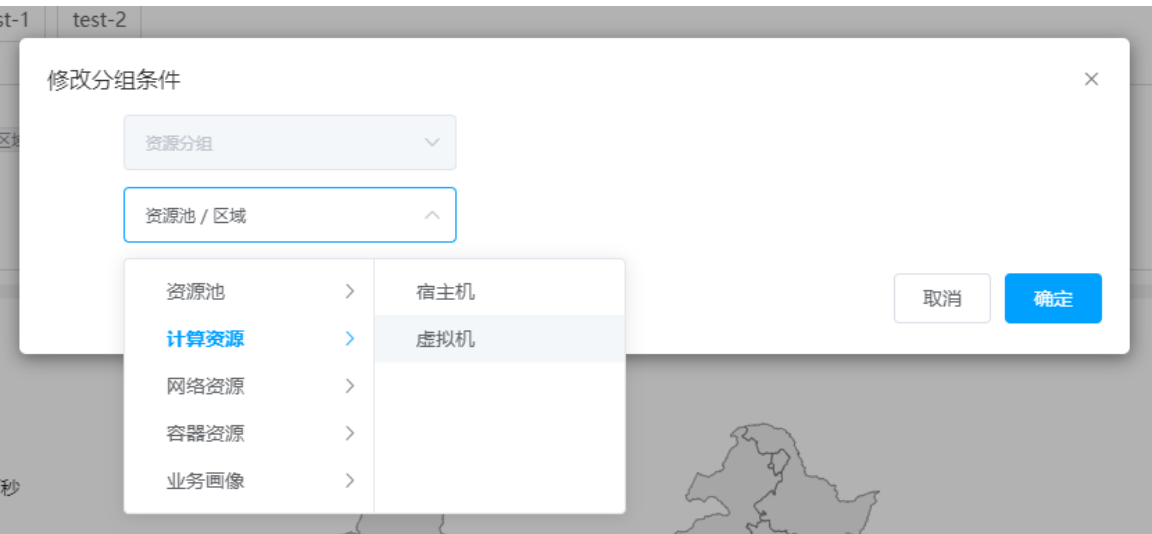
视图列表 查找

新建视图

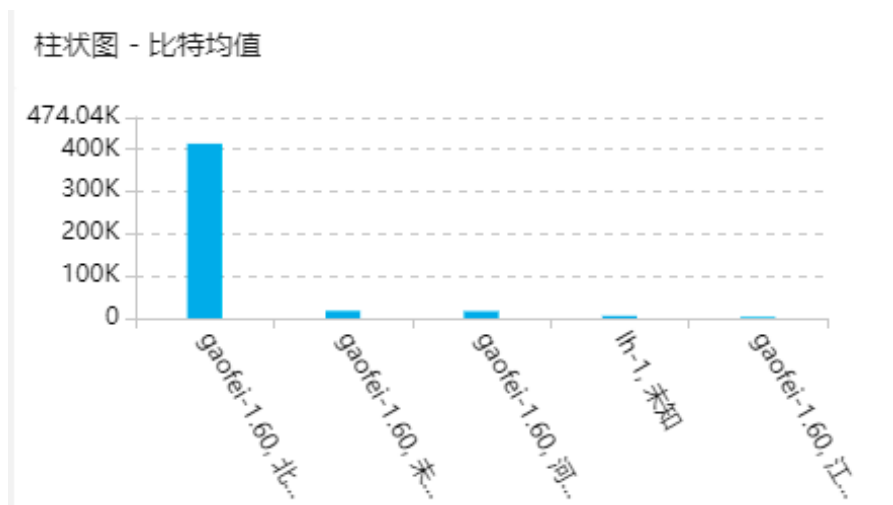
导入视图

视图	描述	权限	来源账号	修改时间	创建时间	操作
流量监控	勿删	所有	x@yunshan.net.cn	2021-01-05 15:32:43	2021-01-05 15:26:39	↩ ● 🗑

- 切换到页面全景图-流量搜索
- 资源分组选择计算资源-云服务器，并查询



- 选择流量排名柱状图，按照上面的步骤加入到流量监控，命名为云服务器-流量Top



- 资源分组选择资源池-区域,并查询。点击流量统计柱形图中第一个柱, 在弹出窗口里选择整体流量

资源知识图谱

区域: [苏州办公室](#) 可用区: [openstack31\(等2个\)](#) 云平台: [openstack31\(等2个\)](#)

租户: [fei\(等5个\)](#) 虚拟机: [host-1-31](#) 虚拟机: [lh-2\(等13个\)](#)

VPC: [Default-zx-os-vpc1\(等12个\)](#) 子网: [zx-os-net1\(等38个\)](#) 路由器: [10.50.150.10_router\(等2个\)](#)

IP地址: [10.244.1.71\(等167个\)](#) 安全组: [default\(等20个\)](#) NAT网关:

负载均衡器: RDS: Redis:

容器集群: [K8S](#) 命名空间: [cattle-system\(等16个\)](#) 容器节点: [k8s-01\(等2个\)](#)

Ingress: [ingress1\(等2个\)](#) 容器服务: [app\(等50个\)](#) 工作负载: [cattle-cluster-agent\(等52个\)](#)

ReplicaSet: [cattle-cluster-agent-58dff56ff8\(等51个\)](#) 容器POD: [cattle-cluster-agent-58dff56ff8-zw2cn\(等...](#) 业务: [1212\(等2个\)](#)

资源组: [111\(等3个\)](#) DHCP网关: [网络-8_lh192-168-88_DHCP\(等29个\)](#) 资源组服务: [null\(等2个\)](#)

负载均衡监听器:

[切换资源分组](#)

区域	可用区	虚拟机
虚拟机	VPC	子网
IP地址	容器集群	命名空间
容器节点	工作负载	容器POD
业务	资源组	

[查看流量曲线](#)

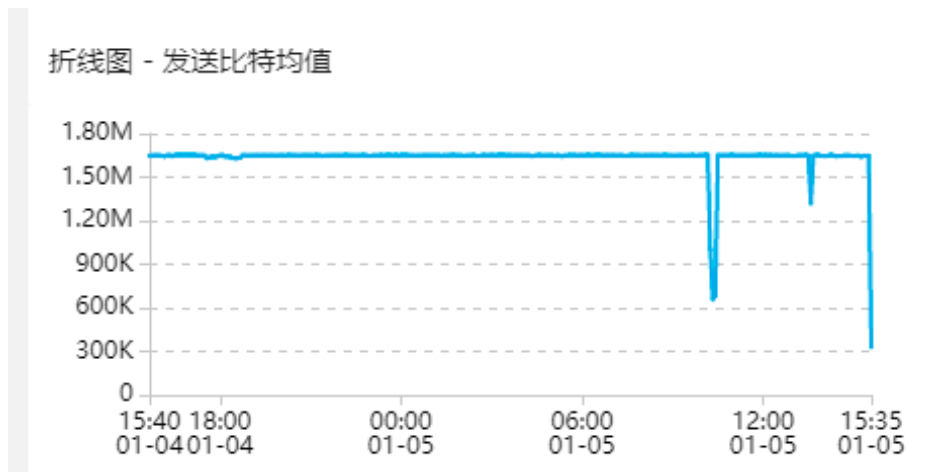
[整体流量](#)

[指标量](#)

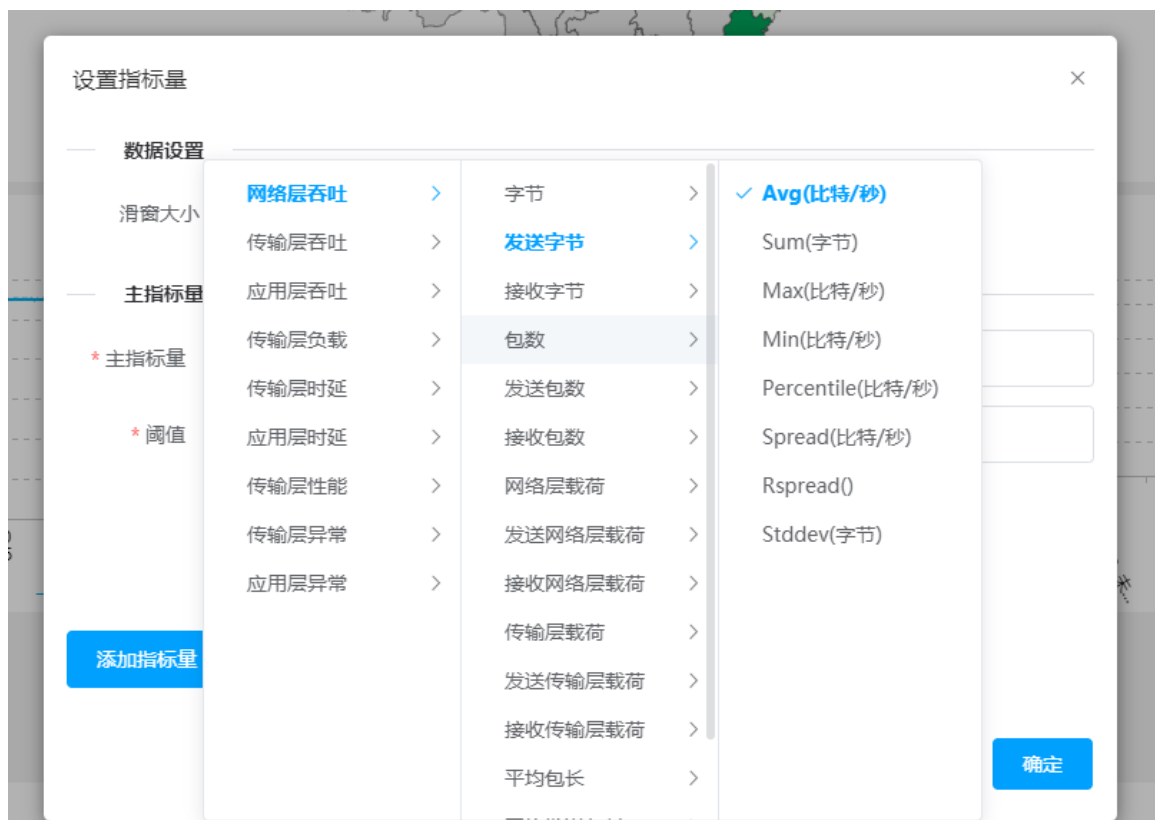
比特均值: 1.59M比特/秒

精准显示 ☐

- 将网络层吞吐中发送比特均值图加入到流量监控中, 命名为区域-接入网络-流量历史



- 为区域-接入网络-流量历史设置主指标量阈值



- 添加告警策略

新建告警策略

×

* 策略名称

监听对象

子视图: 区域-接入网络-流量历史

* 告警等级

☒ 低
☐ 中
☐ 高

* 数据源类型

分钟级数据

* 告警目标

虚拟网络

* 触发条件

发送比特均值

主指标阈值

下限

100

上限

1M

标签

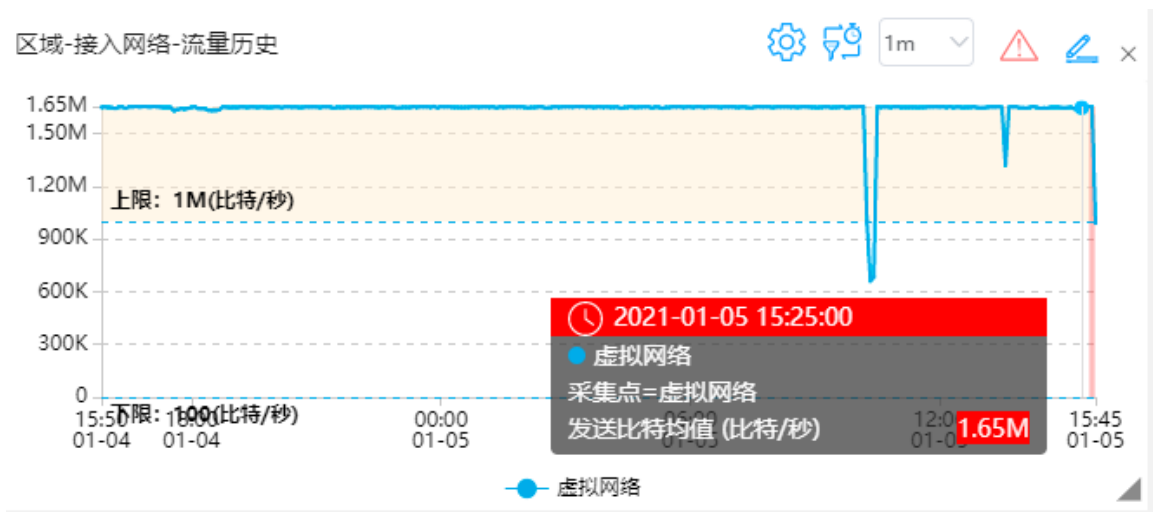
使用Enter键或者逗号分割标签

* 通知邮箱

取消

确定

- 经过一段时间后，可以看到视图中的告警情况：



- 可查看基线阈值，通过历史数据预测当前时间段的流量数据范围。基线阈值相关原理，参考[功能规格](#)

设置指标量

×

数据设置

滑窗大小

1

?

主指标量

* 主指标量

网络层吞吐 / 发送字节 / Avg(比特/...

输入总量计算占比

* 阈值

主指标阈值

下限

100

上限

1M

☒ 基线阈值

算法

ARIMA

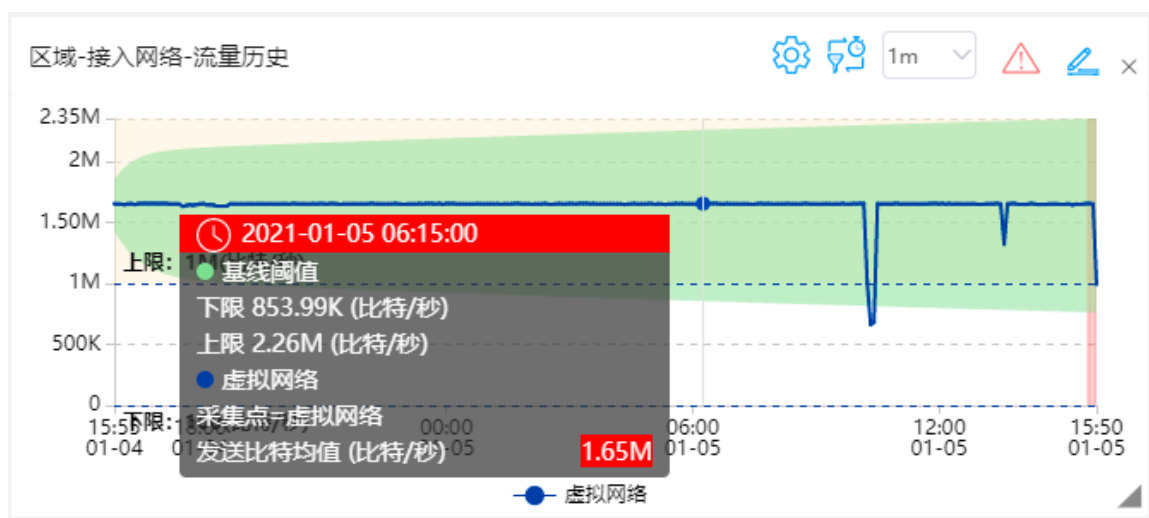
⚠ 流量告警

☒

添加指标量

取消

确定



- 为视图创建一个报表策略



新建报表策略

* 策略名称

* 周期 ☐ 日报表 ☐ 周报表 ☐ 月报表

* 报表格式 ☒ HTML

* 统计粒度 ☐ 每小时 ☐ 每天

* 对象

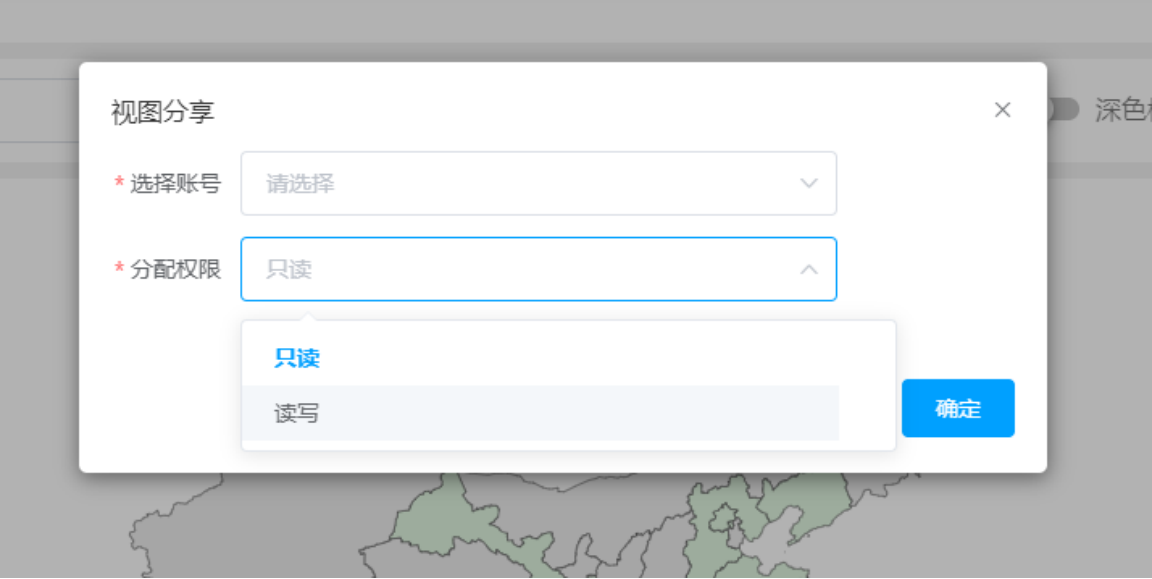
* 推送方式 ☒ 邮件

* 推送邮箱

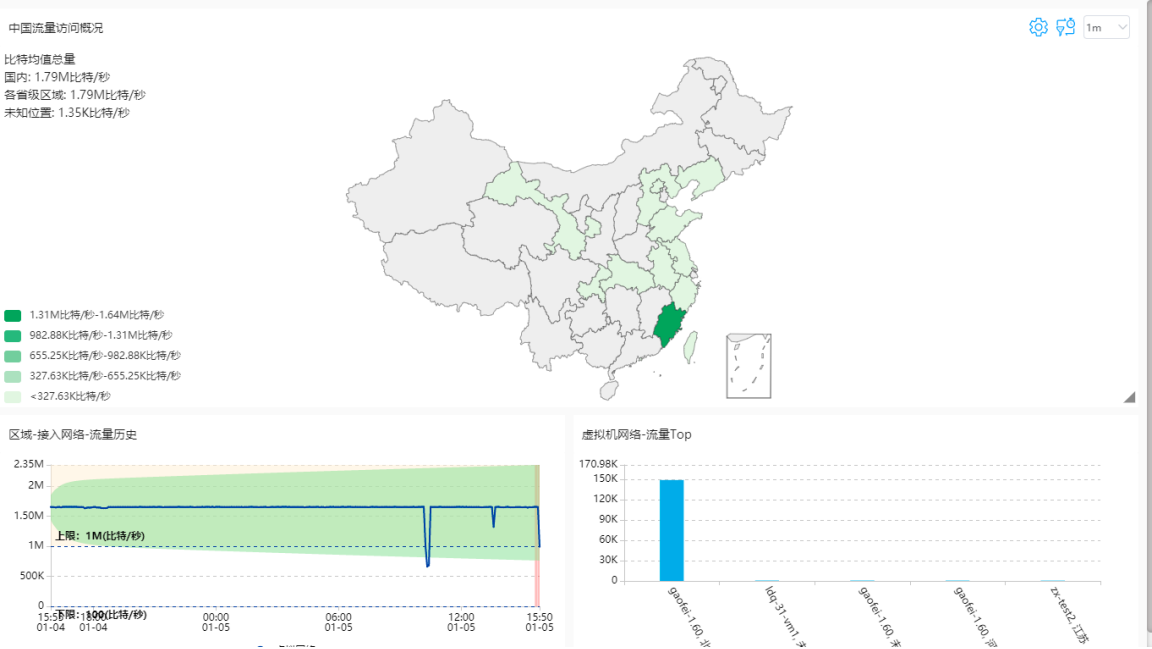
+ 添加推送邮箱 (最多支持10个)

取消 确定

- 可以将设计好的视图分享给其他账户并指定权限



- 经过报表周期后，可以在报表列表中下载对应列表。
- 由此，从功能页根据自己的需求构建了一个简单的视图，并有了告警、报表的能力。
- 此外，在开启时间组件的自动更新情况下，可以将视图全屏化，作为一个简单的监控大屏。



功能规格

规格

项目	规格	修改限制

视图中最多子视图数目	10	可通过配置文件修改
每个账户可以创建视图数目	15	可通过配置文件修改
每个子视图可以创建的告警数目上限	1	不可修改
系统可以创建的报表策略数目上限	100	可以通过配置文件修改
子视图支持秒级数据范围	最近 36 小时	前端配置文件 <code>/var/www/lcweb/public/app_config_user.js</code> 中的 <code>SUPPORT_SECOND_DATA_TIME</code>

授权

每个账户可以访问自己创建的视图，或者访问其他账户分享的视图。

限制

- 暂只支持从功能页中为视图加入子视图
- 加入视图的子视图和原功能页完全独立，只复制相关查询和过滤条件。子视图的条件修改保存不影响功能页内容。

基线阈值原理

深度学习

准确性

- 极值，若函数 $f(x)$ 在 x_0 的一个邻域 D 有定义，且对 D 中除 x_0 的所有点，都有 $f(x) < f(x_0)$ ，则称 $f(x_0)$ 是函数 $f(x)$ 的一个极大值。同理，若对 D 的所有点，都有 $f(x) > f(x_0)$ ，则称 $f(x_0)$ 是函数 $f(x)$ 的一个极小值。
- 最值，设函数 $y=f(x)$ 的定义域为 I ，如果存在实数 M 满足：①对于任意实数 $x \in I$ ，都有 $f(x) \geq M$ ，②存在 $x_0 \in I$ 。使得 $f(x_0)=M$ ，那么，我们称实数 M 是函数 $y=f(x)$ 的最小值，同理，如果存在实数 M 满足：①对于任意实数 $x \in I$ ，都有 $f(x) \leq M$ ，②存在 $x_0 \in I$ 。使得 $f(x_0)=M$ ，那么，我们称实数 M 是函数 $y=f(x)$ 的最大值。
- 由于参数定义域为范围无法确定，因此，只能求出近似最优解。

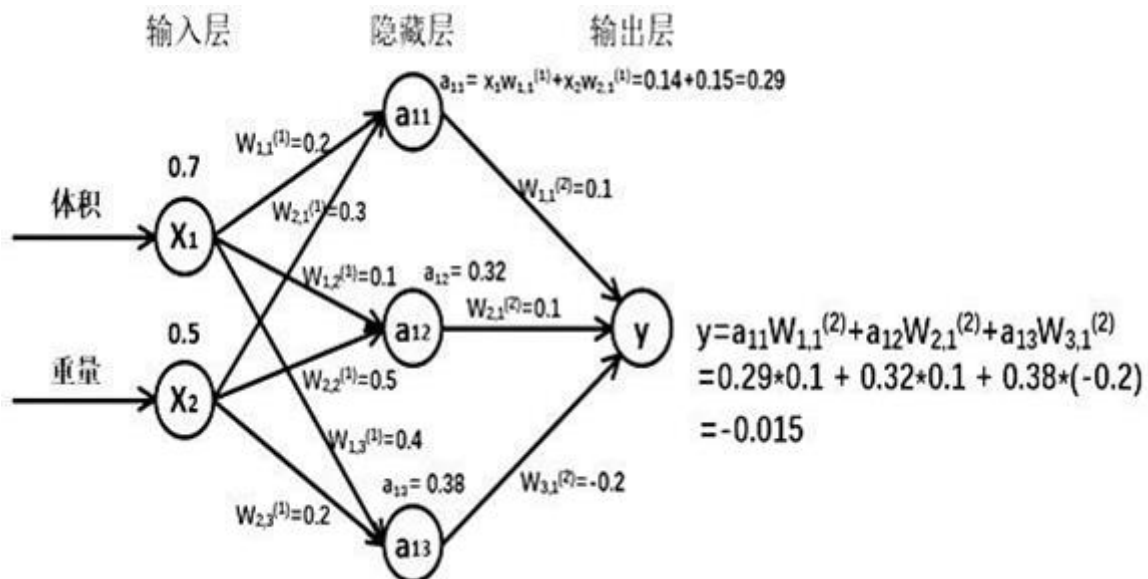
神经网络

神经网络基本框架

- 提取数据集中的特征
- 搭建神经网络结构
- 使用训练数据迭代优化神经网络参数
- 使用训练好的模型预测和分类

向前传播

构造神经网络模型



$$W^{(1)} = \begin{bmatrix} w_{1,1}^{(1)} & w_{1,2}^{(1)} & w_{1,3}^{(1)} \\ w_{2,1}^{(1)} & w_{2,2}^{(1)} & w_{2,3}^{(1)} \end{bmatrix}$$

权重矩阵 $w1$

$$W^{(2)} = \begin{bmatrix} w_{1,1}^{(2)} \\ w_{2,1}^{(2)} \\ w_{3,1}^{(2)} \end{bmatrix}$$

权重矩阵 $w2$

反向传播

术语

- 损失函数，预测值与已知答案的差距
- 训练模型参数，在所有参数上用梯度下降算法，使神经网络模型在训练数据上的损失函数最小

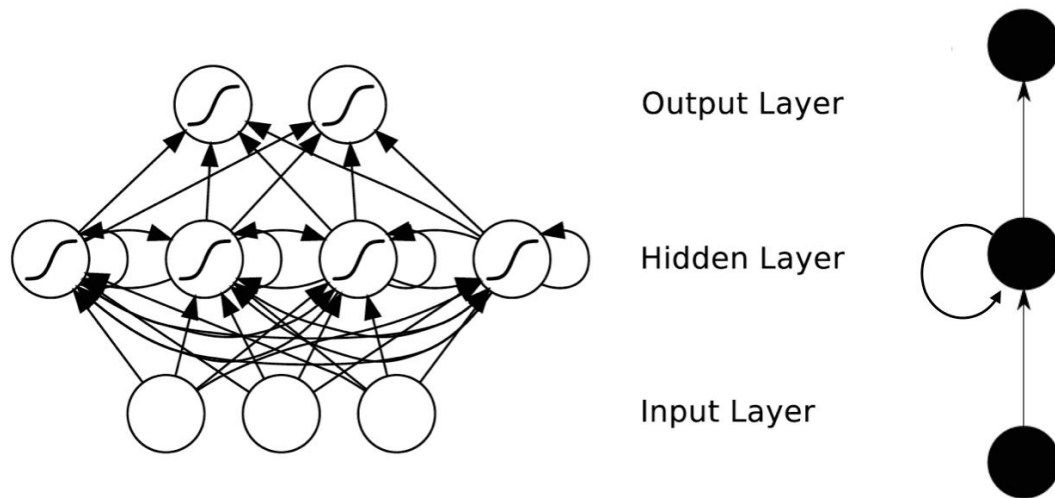
过程

- 步骤 1: 随机初始化权重矩阵
- 步骤 2: 训练集输入模型并得到输出
- 步骤 3: 计算损失函数
- 步骤 4: 按照反向传播训练方法寻找更优权重矩阵
- 回到步骤 2

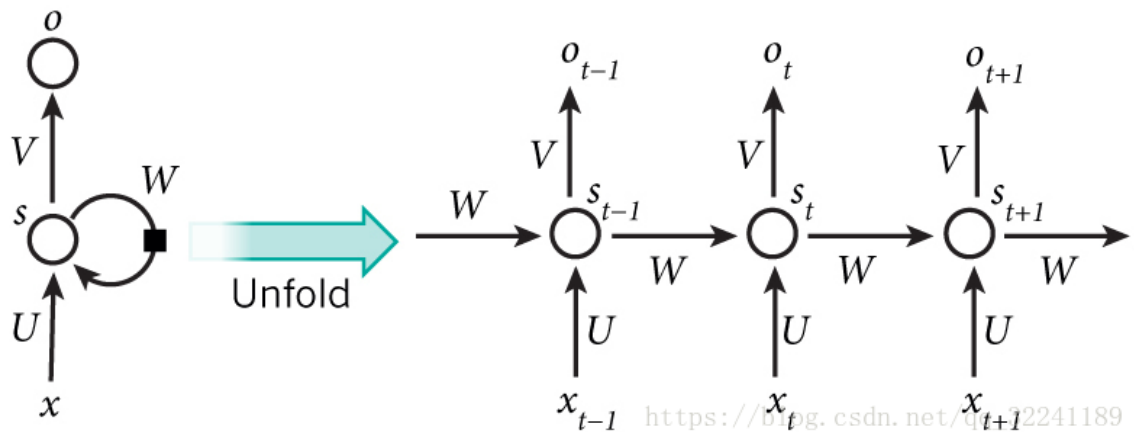
循环神经网络

概述

- 循环神经网络会对前面的信息进行记忆并应用于当前输出的计算中，即隐藏层之间的节点不再无连接而是有连接的，并且隐藏层的输入不仅包括输入层的输出还包括上一时刻隐藏层的输出



- 隐藏层



向前传播

$$h_1 = Ux_1 + Ws_0$$

$$s_1 = f(h_1)$$

$$o_1 = g(Vs_1)$$

- f , 激励函数. 通常是 tanh,relu,sigmoid 等激活函数
- g , 激励函数, 通常是 softmax 也可以是其他

反向传播

$$\nabla U = \frac{\partial E}{\partial U} = \sum_t \frac{\partial e_t}{\partial U}$$

$$E = \sum_t e_t \quad \nabla V = \frac{\partial E}{\partial V} = \sum_t \frac{\partial e_t}{\partial V}$$

$$\nabla W = \frac{\partial E}{\partial W} = \sum_t \frac{\partial e_t}{\partial W}$$

https://blog.csdn.net/qq_32241189

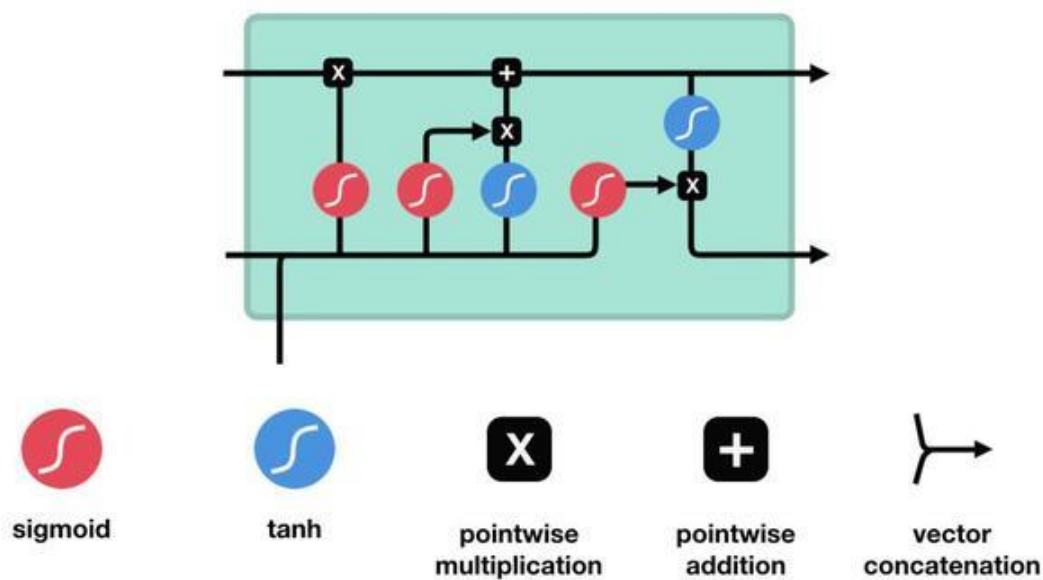
- 根据就向前传播公式可知, 存在 W,U,V 三个权重参数
- 每次输出包含输出值 O_t 和误差值 E_t
- 根据优化算法计算参数梯度

LSTM

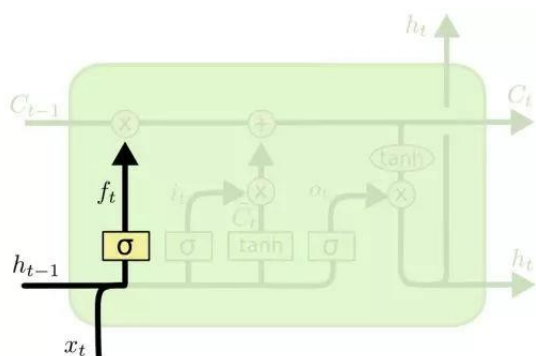
概述

- 长短期记忆网络是循环神经网络的一种变体, 通过精妙的门控制将短期记忆与长期记忆结合起来, 通过门结构来减弱短期记忆影响的演化变体, 其中门结构

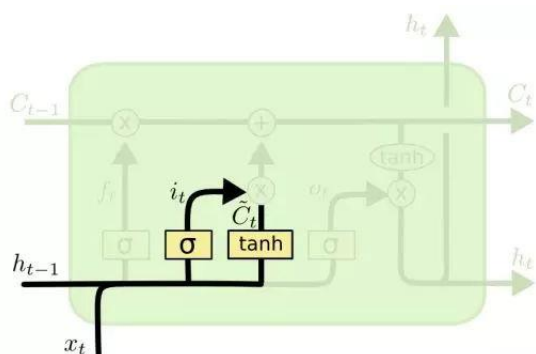
可用来调节流经序列链的信息流，并且一定程度上解决了梯度消失的问题



向前传播

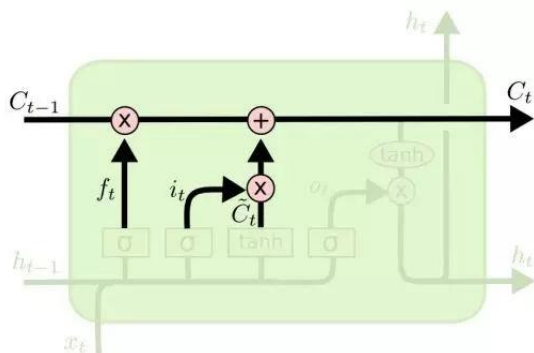


$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

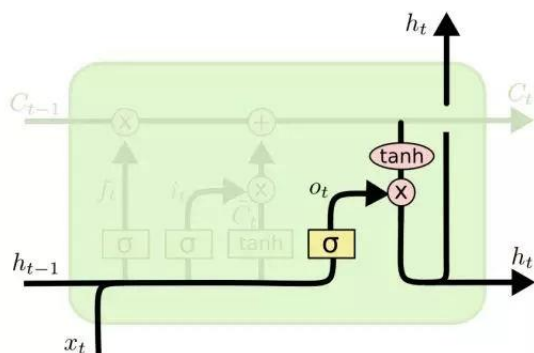


$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$



$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * \tanh(C_t)$$

ARIMA

- ARIMA 模型的全称叫做自回归移动平均模型，全称是(ARIMA, Autoregressive Integrated Moving Average Model)。也记作 ARIMA(p,d,q)，是统计模型(statistic model)中最常见的一种用来进行时间序列预测的模型。

p,d,q

- p-代表预测模型中采用的时序数据本身的滞后数(lags),也叫做 AR/Auto-Regressive 项
- d-代表时序数据需要进行几阶差分化，才是稳定的，也叫 Integrated 项
- q-代表预测模型中采用的预测误差的滞后数(lags)，也叫做 MA/Moving Average 项

公式

$$\hat{y}_t = \mu + \phi_1 * y_{t-1} + \dots + \phi_p * y_{t-p} + \theta_1 * e_{t-1} + \dots + \theta_q * e_{t-q}$$

其中, ϕ 表示 AR 的系数, θ 表示 MA 的系数

模型

- 获取被观测系统时间序列数据;
- 对数据绘图, 观测是否为平稳时间序列; 对于非平稳时间序列要先进行 d 阶差分运算, 化为平稳时间序列;

- 经过第二步处理，已经得到平稳时间序列。要对平稳时间序列分别求得其自相关系数 ACF 和偏自相关系数 PACF，通过对自相关图和偏自相关图的分析，得到最佳的阶数 p 和阶数 q；
- 由以上得到的 d、q、p，得到 ARIMA 模型。然后开始对得到的模型进行模型检验。

告警

告警策略包括视图告警和系统告警两种。其中视图告警在视图->子视图（TopN 折线图、折线图）中创建；系统告警由系统自动创建。

功能使用

告警策略

集中管理平台所有 APP 的告警策略。APP 负责告警策略的创建，创建好的策略，将统一在此处纳管。用户在该功能页查看可查看所有告警策略，包括设定的平台自身的系统告警策略。告警策略中支持设置固定阈值进行触发。

告警策略包括视图告警和系统告警两种。其中视图告警在视图中针对某个子视图创建；系统告警由系统自动创建，仅管理员账号显示系统告警

目前，系统告警策略有：

策略名称	触发信息	指标	默认 阈值	单 位	告警规则
采集器失联	采集器：xx	失联次数	1	次	失联次数大于等于 1 次
采集器丢包	采集器：xx；队列：yy	队列丢包 1 分钟总量	1	个	队列丢包 1 分钟总量大于等于 1 个
采集器异常	采集器：xx；异常：yy	异常事件次数	1	次	异常事件次数大于等于 1 次
采集器 CPU 超限	采集器：xx	CPU 消耗比例 5 分钟谷值	70	%	CPU 消耗比例 5 分钟谷值大于等于 70%
采集器内存超限	采集器：xx	内存消耗比例 5 分钟谷值	70	%	内存消耗比例 5 分钟谷值大于等于 70%
数据节点丢包	数据节点：xx；队列：yy	队列丢包 1 分钟总量	1	个	队列丢包 1 分钟总量大于等于 1 个

					个
数据节点磁盘空间不足	数据节点: xx; 磁盘: yy	磁盘使用空间比例	70	%	磁盘使用空间比例大于等于 70%
数据节点负载高	数据节点: xx	系统负载与 CPU 总数的相对比例 5 分钟谷值	70	%	系统负载与 CPU 总数的相对比例 5 分钟谷值大于等于 70%
数据节点失联	数据节点: xx	失联次数	1	次	失联次数大于等于 1 次
数据节点写入失败	数据节点: xx	数据节点写入失败次数	1	次	数据节点写入失败次数大于等于 1 次
控制器负载高	控制器: xx	系统负载与 CPU 总数的相对比例 5 分钟谷值	70	%	系统负载与 CPU 总数的相对比例 5 分钟谷值大于等于 70%
控制器失联	控制器: xx	失联次数	1	次	失联次数大于等于 1 次
控制器磁盘空间不足	控制器: xx; 磁盘: yy	磁盘使用空间比例	70	%	磁盘使用空间比例大于等于 70%
进程启动	主机名: xx; 进程: yy	进程启动次数	1	次	进程启动次数大于等于 1 次
进程停止	主机名: xx; 进程: yy	进程停止次数	1	次	进程停止次数大于等于 1 次
策略自动删除	资源删除: xx, 关联 yy 删除: zz	策略自动删除触发次数	1	次	策略自动删除触发次数大于等于 1 次
云平台同步异常	云平台: xx, 名称: yy	云平台同步异常触发次数	1	次	云平台同步异常触发次数大于等于 1 次
DeepFlow 授权不足 30 天	授权文件: xx, 可用天数: yy	授权文件可用天数	30	天	最近一个要到期的授权文件可用天数小于等于 30 天
DeepFlow 授权过期	授权文件: xx, 可用天数: yy	授权文件可用天数	0	天	最近一个要到期的授权文件可用天数小于等于 0 天
DeepFlow 预估可用天数不足	剩余金额 (元): xx, 可	剩余金额可用天数	30	天	剩余金额可用天数小于等于 30 天

PCAP 策略：当告警事件结束触发被勾选后，含义为当告警事件结束，PCAP 策略状态关闭，无论之前 PCAP 策略状态为何

Syslog 推送：通过配置推送目的端、Message、推送类型将告警事件以 Syslog 方式发送给用户 * 推送目的端：输入格式为[转发协议]://[日志服务器地址]:[端口]，可选协议为 UDP、TCP，默认为 UDP；地址支持输入域名、IPv4、IPv6 地址，当输入 IPv6 地址时，地址外需要加中括号，例如：UDP://[::]:514；可选端口为 1-65535，默认为 514 * **Message：**支持用 Jinja2 渲染参数，自定义推送内容

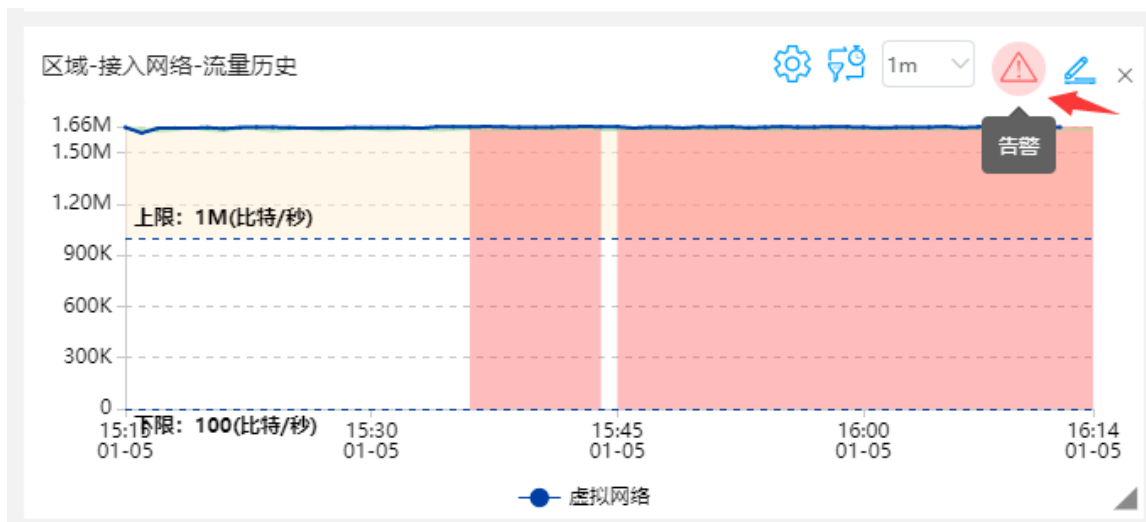
HTTP/Syslog 推送内容中，可自定义的参数有

参数名	释义
policy_id	告警策略 id
policy_name	策略名称
start_time	告警开始时间戳
end_time	告警结束时间戳
start_time_str	告警开始时间，格式为：yyyy-MM-dd HH:mm:ss
end_time_str	告警结束时间，格式为：yyyy-MM-dd HH:mm:ss
policy_app_type	监控对象：1=系统组件，2=视图
policy_sub_type	告警子类型：1=指标量，20=组件状态，21=组件性能，22=自定义删除，23=资源状态，24=平台信息
policy_contrast_type	告警类型：1=阈值告警，2=基线告警
policy_level	告警等级：0=低，1=中，2=高
policy_data_level	数据源：1s，1m，1d
policy_endpoints	推送端点 id
status	告警事件状态：start，end，happened，不可自动恢复类告警状态为 happened
policy_target_uid	系统告警的组件 id，视图告警时此值为告警目标 id
policy_target_name	系统告警的组件名称，视图告警时此值为告警目标名称
policy_go_to	系统组件详情，视图告警时此值为空
policy_target_field	监控指标量
sub_view_id	视图告警的子视图 id，系统告警时此值为-1
sub_view_name	视图告警的子视图名称，系统告警时此值为空
trigger_condition	触发条件
trigger_value	触发值
value_unit	触发值单位

功能使用

创建告警策略

子视图中可以创建视图告警。目前支持时间维度的折线图，以及 TopN 折线图。其中，TopN 折线图告警对超出上限的每条线进行告警 - 步骤一：设置告警阈值：视图->子视图->设置指标量->为需要检测的指标量设置阈值，注意：TopN 折线图可设置上下阈值，折线图仅支持设置上限阈值 - 步骤二：视图->子视图->设置告警->填写下方各项配置



其中，各项配置为：

- * 策略名称：用于标识告警策略的名称
- * 监听对象：固定为子视图，不可修改
- * 告警等级：标识告警策略的严重程度
- * 数据源类型：选择所读取的数据源类型，非日志类折线图/TopN 折线图才有此数据源类型选择
- * 搜索条件展示：不可修改，为创建告警策略时绑定的查询条件。后续子视图修改不会影响告警策略的搜索条件。
- * 告警目标：固定为子视图的统计量
- * 触发条件：固定为子视图的主指标量
- * 上下限：在选择的主指标量大于等于上限或小于等于下限时触发告警策略，（TopN 折线图只支持上限）
- * 标签：用于给告警策略添加文字标签
- * 推送端点：以邮箱/HTTP/Syslog/PCAP 的方式推送告警信息

新建告警策略

×

告警规则：若请求均值大于等于2个/秒时，触发告警

* 主指标量

吞吐 / 请求 / Avg (个/秒)

▼

总量占比

百分比总量

阈值

2

* 策略名称

监听对象

请求均值yanyan (指标d

* 告警等级

☒ 低
☐ 中
☐ 高

* 数据源类型

分钟级数据

▼

* 告警目标

任一满足

▼

?

* 触发条件

请求均值

▼

?

标签

使用Enter键或者逗号分割标签

推送端点

请选择

▼

取消

确定

自定义系统告警策略阈值

- 修改入口：告警策略->系统告警->编辑，不同策略的默认阈值参看告警策略表格所列
- 如下四个策略暂不支持阈值修改：DeepFlow 预估可用天数不足 30 天、DeepFlow 停止服务、DeepFlow 授权不足 30 天、DeepFlow 授权过期
- 列表中，单位为次数类告警策略，统计的为持续时间三分钟内触发的次数，自定义阈值时，生效值仅为 0 次、1 次，阈值超过 1 次时无法触发告警

编辑策略

×

* 策略名称

云平台同步异常

告警规则

若云平台同步异常触发次数大于等于

1

次时，触发告警

推送端点

请选择

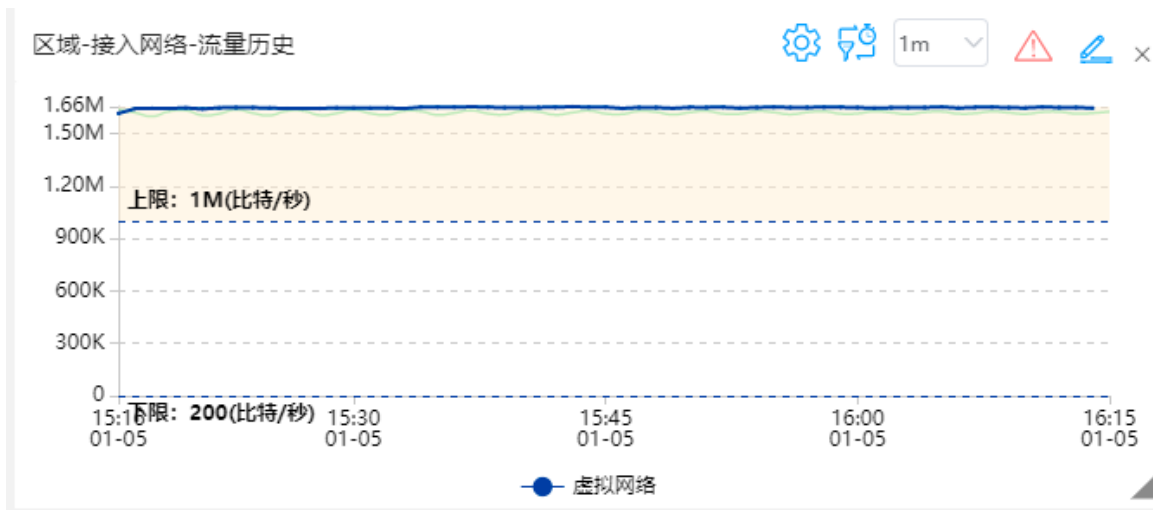
▼

取消

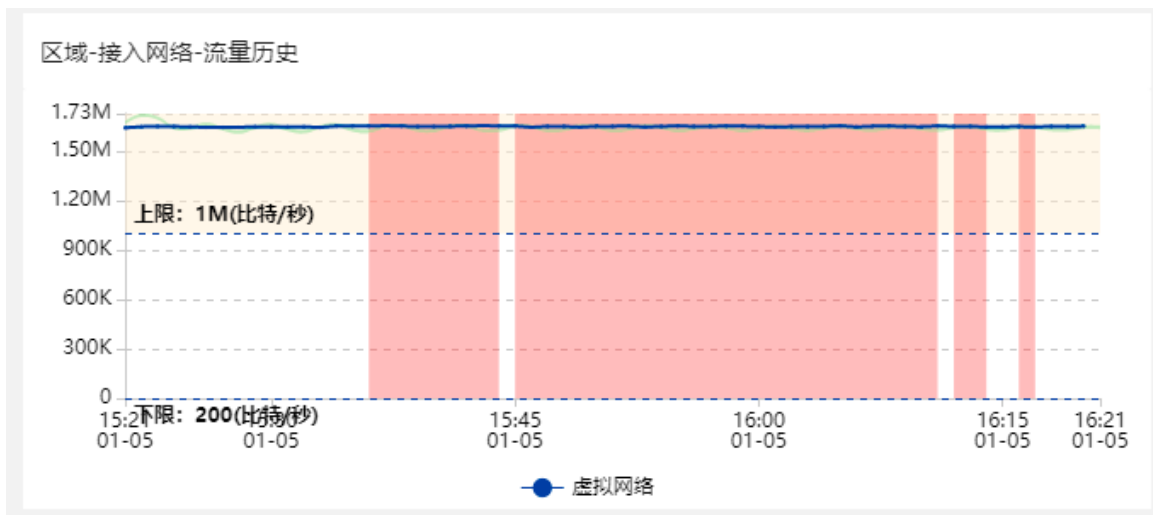
确定

子视图加载告警事件时间段

- 发生告警时，会在告警事件发生或结束时通过告警策略配置的方式发送告警通知。
- 子视图的告警图标在数据范围内，会显示为红色



- 开启展示告警后，会显示时间段内的告警事件



告警策略列表

- 告警策略列表集中呈现系统告警策略与视图告警策略，并可设置告警策略状态、编辑、删除告警策略（系统告警策略不可删除），并可呈现告警策略的触发条件

告警策略列表										
策略名称	监控对象	类型	来源账号	标签	告警等级	告警数量	修改时间	状态	推送端点	操作
超管为租户创建	视图	阈值告警	yanyan@yunsha...		低	0	2021-05-23 14:4...	☒	PCAP策略-租户...	← 🗑
云平台同步异常	系统	阈值告警	--		中	26	2021-05-21 15:5...	☒	邮件推送: jyy-测...	← 🗑
test-jyy	视图	阈值告警	hongliang@yuns...		低	0	2021-05-19 17:5...	☒	--	← 🗑
采集器CPU超限	系统	阈值告警	--		中	0	2021-05-14 18:2...	☒	--	← 🗑
策略自动删除	系统	阈值告警	--		中	0	2021-05-14 18:2...	☒	--	← 🗑
进程停止	系统	阈值告警	--		中	32	2021-05-14 18:2...	☒	邮件推送: jyy-测...	← 🗑
进程启动	系统	阈值告警	--		中	24	2021-05-14 18:2...	☒	邮件推送: jyy-测...	← 🗑
TSDB写入失败	系统	阈值告警	--		中	0	2021-05-14 18:2...	☒	--	← 🗑
采集器内存超限	系统	阈值告警	--		中	0	2021-05-14 18:2...	☒	邮件推送: test-yyl...	← 🗑
数据节点磁盘空...	系统	阈值告警	--		中	3	2021-05-14 18:2...	☒	--	← 🗑

共 30 条 10条/页 < 1 2 3 > 前往 2 页

- 点击单个告警策略对应的事件数量后，跳转至告警事件列表页对该策略的过滤结果

告警事件

告警事件为根据已配置的告警策略，生成的告警事件，目前有不聚合类的告警事件已经聚合类告警事件 * 不聚合类的告警事件为事件开始即结束的事件，一般为不可自动恢复类的告警事件，聚合类告警事件有开始时间与结束时间不相等 * 告警事件列表包括未结束的告警事件统计，新增告警事件统计和告警事件列表 * 告警事件可点击查看详情，其中视图告警可展示事件发生时间段时的子视图并显示对应时间段的流量情况 * 告警事件详情可查看具体触发值、触发条件及跳转至视图或组件的链接

推送端点

- 配置推送端点时，启用策略开启表示告警事件开启时推送，禁用策略开启表示告警事件结束时推送
- 目前支持配置 Email 推送、HTTP 推送、PCAP 推送、Syslog 推送

功能规格

规范

告警事件判定机制处理逻辑

获取系统中的告警策略，每条告警策略按如下方式检查是否有告警事件：

1. 系统每 30 秒检查一次是否触发报警
 - 获取当前时间取整分钟，向前推 **delay** 秒(默认 3 分钟，应对可能的数据落盘延迟，可配置)作为全景图查询的 **end time**
 - 查询的 **end time** 向前推 1 分钟，作为全景图查询的 **start time**
 - 根据子视图对应的全景图 **api** 进行查询，获取结果
 - 与阈值进行对比，超出阈值范围即触发报警

2. 判断是否生成事件
 - 若 1 中结果满足触发条件，且当前没有事件，则当前情况为事件开始，使用 1 中时间窗口的开始时间作为事件的开始时间
 - 若 1 中结果不满足触发条件，且当前有事件，则当前情况为事件结束，使用 1 中时间窗口的开始时间作为事件的结束时间
3. 生成告警事件并推送给用户

报表

通过视图创建报表，每日推送对应的报表结果。

功能使用

功能介绍

视图中可以创建报表，定时推送给用户，以离线下载 html 的格式，记录报表周期内的视图结果。

功能使用

- 视图中创建报表

其中可设置：- 策略名称 - 周期 - 统计粒度 - 对象(固定为当前视图) - 推送方式(邮件，最多支持 10 个邮箱)

新建报表策略

* 策略名称

* 周期 ☐ 日报表 ☐ 周报表 ☐ 月报表

* 报表格式 ☒ HTML

* 统计粒度 ☐ 每小时 ☐ 每天

* 对象 流量监控

* 推送方式 ☒ 邮件

* 推送邮箱

+ 添加推送邮箱 (最多支持10个)

取消 确定

报表的生成策略为每日推送给配置的邮箱。例如周报表，每天推送上一周(如上周二-这周二)的报表。报表从起点时间的当天 0 点，到结束时间的当天 24 点。每次报表策略的修改，以生成报表前的最后一次为准，并从当天开始影响后续生成的报表，对之前的报表不会有影响。且报表邮件中包含报表附件，点击可下载

- 查看报表策略列表

在报表-策略中查看所有的报表策略列表

通过此功能对报表策略进行统一的管理，可实现报表策略的查看、修改、删除等操作。

报表策略							
按策略名称搜索							
策略名称(报表数量)	周期	对象	推送邮箱	创建时间	修改时间	禁/启用	操作
秀梅(已生成0个)	日报表		Z@qq.com	2020-10-12 14:59:20	2020-10-12 14:59:20		
321(已生成20个)	日报表		x@123.df	2020-09-24 14:09:58	2020-09-24 14:09:58		

每页行数: 101-2 共 2 条< 1 / 1 >

点击策略名称，可以跳转到报表列表页查看对该策略的过滤结果。

- 查看报表列表

在报表-列表中查看所有生成的报表结果，按生成时间逆序排列(最近生成的在最前)。可以下载离线 html 包，或者删除无用的报表。报表中子视图支持下载 CSV 数据。

< 报表策略列表

根据平台所定义策略产生所有报表列表，支持ZIP和CSV两种下载格式。

报表策略: 321

报表下载					
按报表名称搜索					
报表名称	策略名称	周期	视图名称	生成时间	操作
[DeepFlow 报表]321-日报-2020-...	321	日报表	222	2020-10-12 06:00:45	
[DeepFlow 报表]321-日报-2020-...	321	日报表	222	2020-10-10 06:00:45	
[DeepFlow 报表]321-日报-2020-...	321	日报表	222	2020-10-09 06:00:45	
[DeepFlow 报表]321-日报-2020-...	321	日报表	222	2020-10-08 06:00:45	
[DeepFlow 报表]321-日报-2020-...	321	日报表	222	2020-10-07 06:00:45	
[DeepFlow 报表]321-日报-2020-...	321	日报表	222	2020-10-06 06:00:45	
[DeepFlow 报表]321-日报-2020-...	321	日报表	222	2020-10-05 06:00:45	

功能规格

规格

项目	规格	修改限制
报表策略上限	100	配置文件 /etc/report.yaml 中 policy_num_max，需要重启 report 进程

资源

资源管理

自动或手动同步云网中的资源信息，是全景图十余个维度搜索能力的基石。

功能使用

概念映射

资源管理将不同厂商的云平台进行抽象和整理，适配统一的概念，DeepFlow 平台的其他应用都基于统一的概念来分析和呈现数据。

目前已支持和学习的云平台资源包括：

- 公有云：AWS、腾讯公有云、阿里公有云、华为公有云、青云公有云、微软公有云
- 开源私有云：OpenStack、ZStack
- 闭源私有云：vSphere、华为云、青云私有云、腾讯专有云、阿里专有云、金山银河云、2Cloud/NSP
- 容器：Kubernetes

公有云

云平台	AWS	腾讯公有云	阿里公有云	华为公有云	青云公有云	微软公有云
租户	无	无	无	无	无	无
区域	区域 (Region)	区域	区域	区域	区域	区域
可用区	可用区 (AZ)	可用区	可用区	可用区	可用区	可用区
云服务器	实例 (Instance)	实例	实例	弹性云服务器	主机	虚拟机
宿主机	无	无	无	无	宿主机	无
VPC	VPC	私有网络	专有网络	虚拟私有云	VPC 网络	虚拟网络
子网	子网 (Subnet)	子网	交换机	子网	私有网络	子网
路由器	路由表 (RoutingTable)	路由表	路由表	虚拟私有云	路由表	路由表
DHCP 网关	无	无	无	无	无	无
公网 IP	弹性 IP (Elastic IP)	公网 IP、弹性 IP	弹性公网 IP	弹性公网 IP	公网 IP	公共 IP
NAT 网关	NAT 网关	NAT 网关	NAT 网关	NAT 网关	NAT	NAT

关 负载均衡器	负载均衡器 (LoadBalancer)	关 负载均衡	关 负载均衡	关 弹性负载均衡器	网关 负载均衡	网关 无
对等连接	对等连接 (PeerConnection)	对等连接	无	对等连接	无	无
云企业网	无	无	云企业网	无	无	无
安全组	安全组 (SecurityGroup)	安全组	安全组	安全组	安全组	安全组
云数据库 RDS	无	无	云数据库 RDS	无	无	无
云数据库 REDIS	无	无	云数据库 Redis	无	无	无

开源私有云

云平台	OpenStack	ZStack
租户	用户 (User)	账户
区域	无	无
可用区	资源池	资源池
云服务器	实例 (Instance)	云主机
宿主机	云主机 (Host)	物理机
VPC	项目 (Project/Tenant)	VPC 路由器
子网	网络 (Network)	VPC 网络
路由器	路由 (Router)	VPC 路由器
DHCP 网关	DHCP 网关	无
外网 IP	浮动 IP (Floating IP)	弹性 IP 地址
NAT 网关	无	无
负载均衡器	无	无
对等连接	无	无
云企业网	无	无
安全组	安全组 (SecurityGroup)	无
云数据库 RDS	无	无
云数据库 REDIS	无	无

闭源私有云

云平 台	vSphere	华为云	腾讯 专有 云	阿里 专有 云	青云 私有 云	金山银河云	NSP
租户	无	用户	无	无	无	无	用户
区域	无	无	区域	区域	区域	无	无
可用 区	资源池	资源池	可用 区	可用 区	可用 区	资源池	AZ
云服 务器	虚拟机 (VM)	虚拟机	实例	实例	主机	实例	虚拟 机、包 括虚拟 机形态 的负载 均衡器
宿主 机	主机 (Host)	宿主机	母机	无	宿主 机	云主机	宿主机
VPC	数据中心 (DataCenter)	项目	私有 网络	专有 网络	VPC 网络	项目 (Project/Tenant)	逻辑网 络
子网	端口组 (PortGroup)	网络	子网	交换 机	私有 网络	网络 (Network)	逻辑交 换机
路由 器	无	路由器	路由 表	路由 表	路由 表	路由 (Router)	逻辑路 由器、 虚拟网 关、虚 拟防火 墙
DHCP 网关	无	无	无	无	无	无	无
外网 IP	虚拟机 IP	Floating IP	公网 IP、 弹性 IP	弹性 公网 IP	公网 IP	浮动 IP (Floating IP)	公网 IP
NAT 网关	无	无	NAT 网关	NAT 网关	NAT 网关	无	无
负载 均衡 器	无	无	负载 均衡	负载 均衡	负载 均衡	无	无
对等	无	无	对等	无	无	无	无

连接			连接				
云企	无	无	无	云企	无	无	无
业网				业网			
安全	无	安全组	安全	安全	安全	安全组	无
组			组	组	组	(SecurityGroup)	
云数	无	无	无	无	无	无	无
据库							
RDS							
云数	无	无	无	无	无	无	无
据库							
REDIS							

容器

容器平台	Kubernetes
租户	无
区域	无
可用区	集群 (Cluster)
VPC	集群 (Cluster)
子网	节点 POD 网段、服务的 ClusterIP 网段
外网 IP	节点 (Node) IP
集群	集群 (Cluster)
命名空间	命名空间 (Namespace)
节点	节点 (Node)
POD	POD
工作负载	Workload/Controller
服务	服务 (Service)
INGRESS	Ingress

页面操作

所有表格信息均支持 CSV 下载与列选择，部分资源信息支持设置别名。

- 资源池
 - 支持录入区域，支持修改区域的经纬度，用于在全景图中绘制正确的区域地图
 - 支持录入可用区，支持在录入可用区时与区域关联
 - 支持录入云平台，用于资源对接；支持修改云平台配置
 - 以下云平台不支持额外对接附属容器集群

- ZStack
 - NSP
 - Kubernetes
 - 青云 QKE
 - F5
 - 平安金融云
 - 招行 ACS
- 计算资源
 - 支持录入云服务器与网卡，支持指定区域、可用区、云平台、VPC、类型、宿主机
 - 支持录入宿主机与网卡，支持指定区域、可用区、云平台、管理 IP、类型
 - 点击云服务器名，可查看云服务器虚接口的采集和分发速率
- 网络资源
 - 支持录入 VPC，支持将录入的 VPC 与区域关联
 - 支持录入子网，支持将录入的子网与区域、可用区、VPC 关联
 - 对于录入的子网，支持指定子网的类型（外网/内网）和网段（多个 IPv4/IPv6 CIDR）
 - 支持查看路由器的路由表
- 网络服务
 - 支持查看安全组、NAT 网关、负载均衡器的策略表
- 容器资源
 - 支持查看服务、Ingress 的策略表
- 其他资源
 - 支持录入物理网元并指定其类型和区域
 - 支持录入物理采集点，采集点支持镜像、分光、sFlow、NetFlow
 - 支持录入物理链路，关联两侧的物理网元和采集点
 - 支持录入资源组、云平台的自定义图标

为同步资源指定区域

DeepFlow 采集器仅能与区域内的从控制器和数据节点通信，因此区域是 DeepFlow 中资源的封闭单元。无论手动录入或自动同步，都需要考虑为资源设置合理的区域属性。

为同步资源指定控制器

混合云环境下（阿里云混合云方案、线上 POC 公有云中的 K8s 资源），通过主/备控制器对接不同云平台资源的架构不合理；需要由从控制器对接某些平台资源。

- 仅页面录入的云平台支持指定资源同步控制器

自动同步区域信息

对于公有云，通过自动同步可学习到资源在公有云运营商上的区域信息。对于私有云和容器，不指定区域信息时自动加入系统默认的区域。

修改区域信息

进行资源同步时，可通过指定区域属性，将所有可用区与指定的区域关联，配置方法详见[部署扩容](#)。

警告： 注意： 修改已同步资源的区域属性，将会对相关资源重新同步，会导致对应的流量历史统计数据无法查询。

典型场景

录入子网

警告： 所有未同步到资源管理中的 IP 地址都会被系统认为是 Internet 地址，需要监控但无法自动同步的网段需要在网络资源-子网信息中录入，以避免系统将其识别为 Internet 地址。

下载 CSV

点击每个列表的左上角的下载按钮，可全量下载当前列表的所有数据并以 CSV 格式输出。

功能规格

规格

资源	默认限制	修改限制
云平台	4 种	不可修改
云平台附属 Kubernetes	10 个	可修改 /var/www/lcweb/public/app_config_user.js 的 DOMAIN_ATTACHED_K8S_COUNT
物理采集点	250 个	不可修改
租户	1000 个	不可修改
区域	100 个	不可修改
可用区	100 个	不可修改
VPC	1000 个	不可修改
子网	5000 个	不可修改
云服务器	20000 个 (vSphere 最 大 5000 个)	不可修改
宿主机	2000 个	不可修改

	(vSphere 最大 500 个)	
安全组	200 个	不可修改
单安全组的规则	200 条	不可修改
对等连接	100 个	不可修改
单 NAT 网关规则	200 条	不可修改
单负载均衡器规则	200 条	不可修改
单负载均衡器规则	200 条	不可修改
容器集群	100 个	不可修改
容器节点	1000 个	不可修改
容器 POD	5000 个	不可修改

限制

类型	云平台	支持版本
公有云	AWS	N/A
公有云	阿里公有云	N/A
公有云	腾讯公有云	N/A
公有云	华为公有云	N/A
公有云	青云公有云	N/A
公有云	微软公有云	N/A
开源私有云	OpenStack	keystone: v2/v3、nova: v2.x、neutron: v2.0
开源私有云	ZStack	企业版 3.1.1、企业版 3.4.0、混合云版 3.5.0.1
闭源私有云	vSphere	5.1、5.5、6.0、6.5
闭源私有云	华为云	FusionSphere 6.3.1、ManageOne 6.3.1/8.0.0、华为云 stack 6.5.1
闭源私有云	腾讯专有云	3.6.0
闭源私有云	阿里专有云	3.13

闭源私有云	青云私有云	API Version 1
闭源私有云	金山银河云	Keystone v2、neutron v2.0
闭源私有云	2Cloud/NSP	NSP 5.5.6
容器	Kubernetes	Kubernetes v1.12、Kubernetes v1.13、Kubernetes v1.16

其他限制： - 容器环境中支持同步 Deployment、StatefulSet、ReplicationController、DaemonSet 四种工作负载 - 容器环境中仅支持同步 Node Port 和 Cluster IP 两种服务 - 不支持同步 L7 类型的负载均衡监听器规则 - 不支持同步华为公有云的路由表，目前是在每个 VPC 下面自动生成了一个路由表 华为公有云 API 暂时不支持获取路由表 - Workload-V 类型采集器同步资源信息时，如果将云服务器关机，无法同步云服务器状态变化，会一直处于运行状态 - KVM 类型采集器同步资源信息时，无法感知和同步宿主机、虚拟机的名称变化

租户权限

租户登录 - 与 VPC 和命名空间相关的资源：只显示授权的资源列表信息 - 其他：显示全部资源

业务画像

业务画像可将 VPC 或者子网的资源通过流量进行梳理，得到这组资源内部的应用服务依赖和流量访问关系。

功能使用

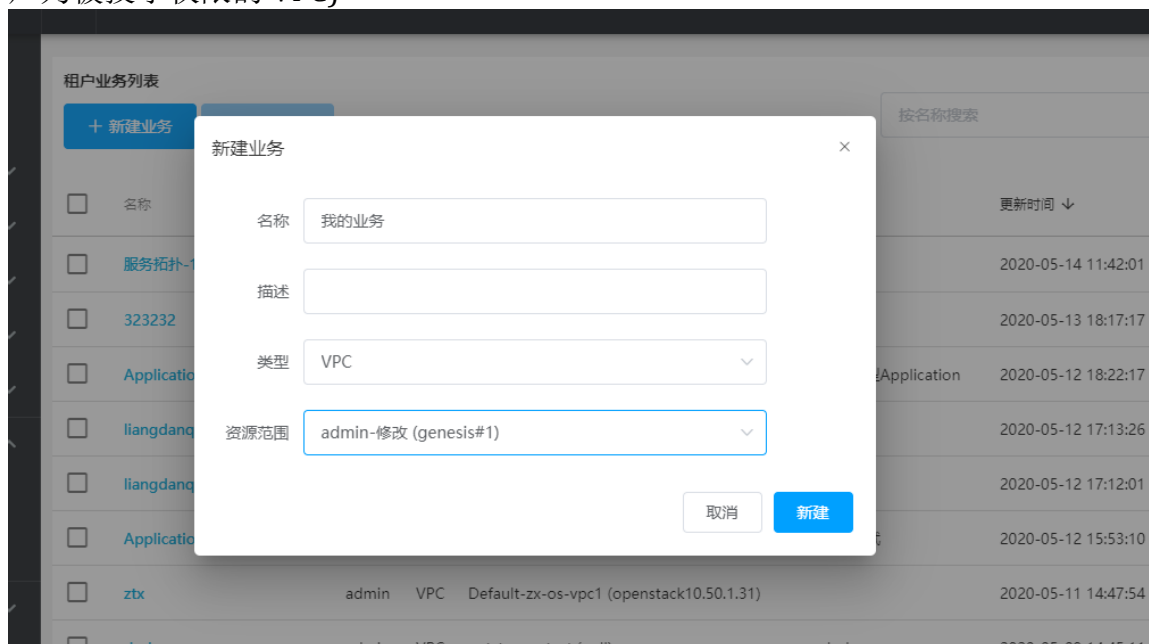
功能介绍

业务画像可将 VPC 或者子网的资源通过流量进行梳理，得到这组资源内部的应用服务依赖和流量访问关系，并可将这个关系通过业务、资源组、服务依赖保存下来，在全景图中查询对应的流量。

功能使用

- 创建业务

业务类型可以是 VPC 或者子网，对应的资源范围选择系统中的 VPC 和子网(用户为被授予权限的 VPC)



- 业务列表页可以完成对业务的修改和删除

租户业务列表							
☐ 名称 ↑		用户	类型	资源范围	描述	更新时间 ↓	操作
☐	服务拓扑-1589427610540	admin	子网	10.50.0.0/16		2020-05-14 11:42:01	 
☐	323232	admin	VPC	Datacenter (vsphere-172.29.1.5)	212112	2020-05-13 18:17:17	 
☐	Application-LAI-VPC	admin	VPC	kubernetes#3 (kubernetes#3)	VPC类型Application	2020-05-12 18:22:17	 
☐	liangdang-qing-k8s-bj	admin	VPC	kubernetes#3 (kubernetes#3)	k8s-bj	2020-05-12 17:13:26	 
☐	liangdang-qing-k8s-sz	admin	VPC	kubernetes#1 (kubernetes#1)	k8s-sz	2020-05-12 17:12:01	 
☐	Application-LAIYUAN	admin	子网	172.31.1.0/24	应用测试	2020-05-12 15:53:10	 
☐	ztx	admin	VPC	Default-zx-os-vpc1 (openstack10.50.1.31)		2020-05-11 14:47:54	 

- 进入业务详情页后，可以人工为业务中资源创建资源组

< 返回列表 / 323232

[+ 新建资源组](#) [批量删除](#)

☐	图标	名称	类型	资源	修改时间 ↓	操作
☐		Internet	数据中心出口设备资源组		2020-05-14 13:54:38	
☐		wq	虚拟机资源组	VM1-ZL,tridentvm,VM2-ZL	2020-05-14 10:58:02	
☐		23	虚拟机资源组	VM1-ZL,tcpreplay2-100-2,tridentvm,VM2-ZL,deepflow1...	2020-05-13 18:29:33	
☐		123	网段资源组	10.50.100.72	2020-05-13 18:22:43	
☐		12	网段资源组	10.50.101.29	2020-05-13 18:20:43	

每页行数: 10 1-5 共 5 条 < 1 / 1 >

新建资源组

名称

类型

图标

关键服务列表

[添加](#)

服务

操作

虚拟机资源组

网段资源组

容器服务后端POD组资源组

容器服务资源组

负载均衡后端主机资源组

无可用数据

每页行数: 10 < 1 / 1 >

[取消](#) [新建](#)

可以根据资源的实际属性，选择对应类型和图标。目前支持：

- 云服务器资源组，可选择业务内的若干云服务器。

- 网段资源组(子网类型业务只支持该类型资源组)，支持 IP 地址、IP 范围、IP 段的形式定义的网段资源。
- 容器服务资源组，可从业务范围中的容器服务中选择。
- 容器服务后端 POD 资源组，可从选择的容器服务的对应后端工作负载中选择。
- 负载均衡后端主机资源组，可从选择的负载均衡器的对应负载均衡策略中进行选择。

建议： 注意，云服务器、容器服务、负载均衡均从资源管理中读取，并按对应的 VPC 进行过滤。5.6.0 中暂未支持负载均衡器资源组，可通过网段资源组来实现对应需求，将负载均衡器对外服务 IP 和关注的对内转发 IP 加入到对应网段资源组，并设置对应的关键服务列表。当负载均衡器为应用层负载均衡器时，注意源 IP 可能未发生变化，流量中会直接显示源->负载均衡器后端主机资源组。需要人工加入服务依赖来定义资源组之间的服务关系。

5.6.0 新增了关键服务列表，用于标记资源组关心的相关服务内容。其中，

- 云服务器和网段类型资源组可以手动添加

- 容器服务类型资源组自动获取服务对外端口列表
- 容器服务后端 POD 资源组自动获取工作负载端口列表
- 负载均衡后端主机资源组自动从负载均衡策略中获取端口列表

资源组之间可以通过服务依赖进行关联，描述资源组之间的依赖关系。

新建服务依赖

名称

源资源组

请选择

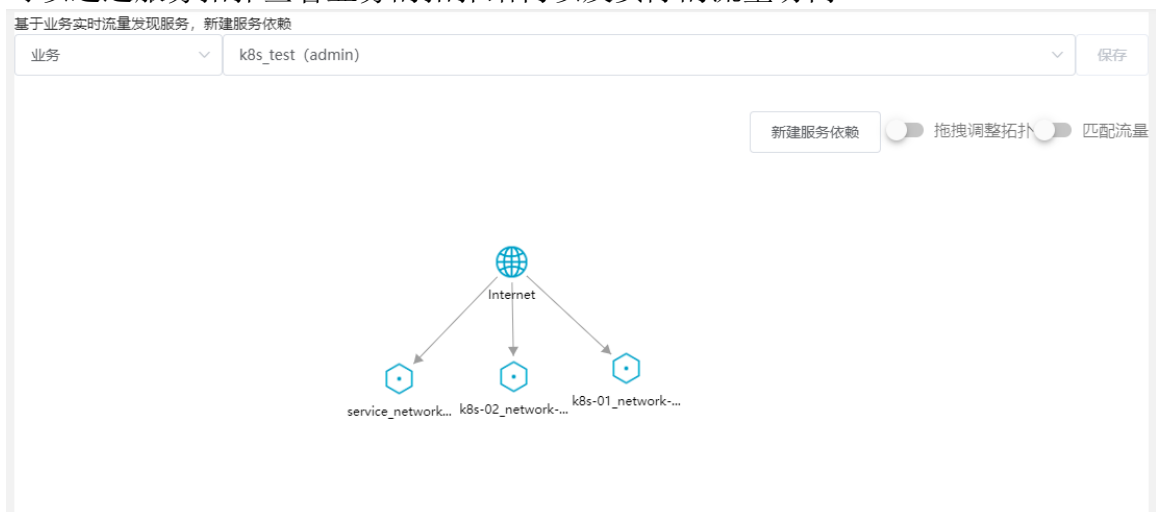
目的资源组

请选择

取消

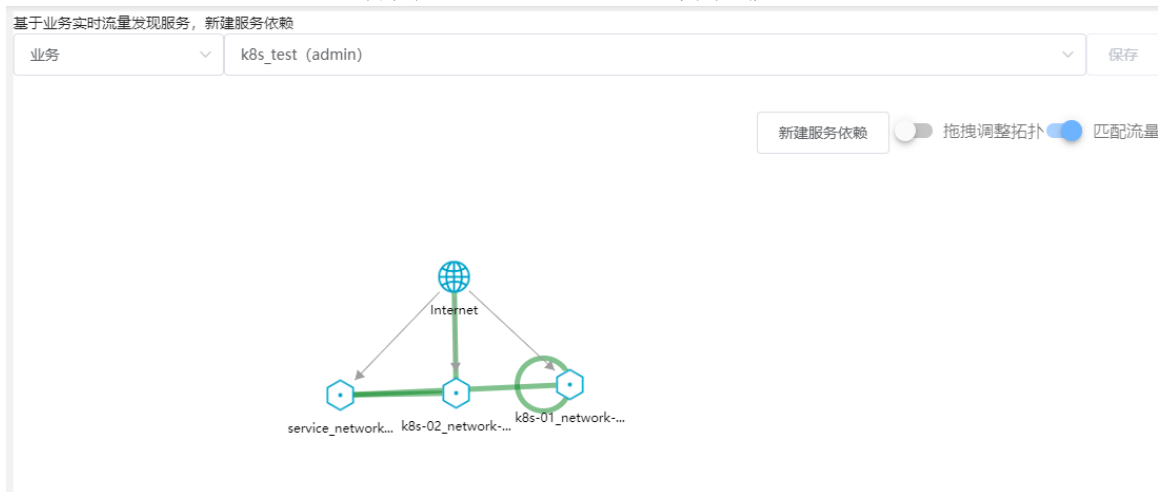
新建

- 可以通过服务拓扑查看业务的拓扑结构以及实际的流量访问



- 支持查看业务，也同时支持 VPC 和子网，会自动生成一个临时业务，可在离开页面时保存业务。
- 服务拓扑可查看业务的服务依赖拓扑

- 可通过匹配流量开关，查看真实的流量叠加在服务依赖拓扑上的走向。



- 通过点击图中节点，可以查看对应的节点上的流量信息

资源组 k8s-01_network-Ygei 流量详情

匹配关键服务

服务端IP流量TOP100
统计时间范围内节点中服务端IP的TCP和UDP流量

输入过滤条件，回车完成输入 查询 加入资源组

☐	服务端IP	服务端端口	服务端设备	所属资源组	采集点	客户端请求流量	服务端回复流量	总流量
☐	10.244.0.21	TCP 36578	nginx-ingress-controller-68cff6666-fxcw22	k8s-01_network-Ygei	虚拟网络	95.02MB	0.00B	95.02MB
☐	10.244.0.15	TCP 35202	coredns-576cbf47c7-s4mkd	k8s-01_network-Ygei	虚拟网络	47.41MB	0.00B	47.41MB
☐	10.244.0.14	TCP 57324	coredns-576cbf47c7-p2dhc	k8s-01_network-Ygei	虚拟网络	47.41MB	0.00B	47.41MB
☐	10.244.0.21	TCP 10254	nginx-ingress-controller-68cff6666-fxcw22	k8s-01_network-Ygei	虚拟网络	7.91MB	8.75MB	16.66MB
☐	10.244.0.15	TCP 8080(http-alt)	coredns-576cbf47c7-s4mkd	k8s-01_network-Ygei	虚拟网络	4.07MB	4.06MB	8.14MB
☐	10.244.0.14	TCP 8080(http-alt)	coredns-576cbf47c7-p2dhc	k8s-01_network-Ygei	虚拟网络	4.00MB	4.08MB	8.08MB

每页行数: 10 1-6 共 6 条 < 1 / 1 >

服务实时流量TOP100
统计时间范围内节点中服务端IP分组的出入流量

- 通过流量连线 tooltip，可以点击查看对应的服务依赖上的流量信息

k8s-02_network-0lvB -> service_network-GsQL 流量详情

流量Top100

统计两个资源组(包括未分组资源)之间的TCP和UDP流量

匹配服务依赖

查询

输入过滤条件，回车完成输入

源资源组	目的资源组	采集点	服务端IP	客户端请求流量	服务端回复流量	总流量	操作
k8s-02_network-0lvB	service_network-GsQL	虚拟网络	TCP 443(https)	44.16MB	273.69MB	317.85MB	查看详情

每页行数: 10

1-1 共 1 条

< 1 / 1 >

并可进一步查看两个资源组之间流量的详情

k8s-02_network-0lvB -> service_network-GsQL 流量详情

IP流量详情TOP100

按服务端IP和客户端IP统计时间范围内业务中流量

查询

输入过滤条件，回车完成输入

服务端IP	服务端设备	服务端IP	客户端IP	客户端设备	源出流量	源入流量	源端总流量
10.96.0.1		TCP 443	10.244.1.32	cattle-cluster-agent-58dff56ff8-zw2cn	44.19MB	273.87MB	318.06MB

每页行数: 10

1-1 共 1 条

< 1 / 1 >

服务拓扑中的未分组服务端资源组包含：业务中提供服务但未被加入任何业务内资源组的资源；未分组客户端资源组包含：业务中作为客户端但未被加入任何业务内资源组的资源。当不存在这样的资源时，不显示对应的未分组资源组。由于业务画像主要对服务进行划分和依赖分析，因此未分组客户端资源组没有流量详情，但可以通过未分组客户端资源组到其他资源组之间的流量连线查看有哪些客户端 IP。

- 资源组(Internet 资源组除外)流量详情中，可以人工选择服务 IP 组成新的资源组，并可选择是否从现有资源组中移除。

未分组服务端资源 流量详情

服务端IP流量TOP100
统计时间范围内节点中服务端IP的TCP和UDP流量

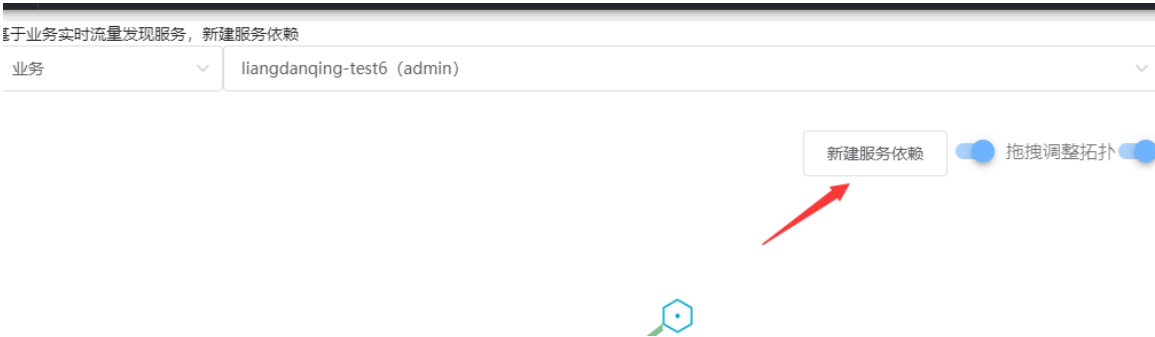
输入过滤条件，回车完成输入

查询 加入资源组

	服务端IP	服务端端口	服务端设备	所属资源组	采集点	客户端请求流量	服务端回复流量	总流量
☒	10.96.0.1	TCP 443(https)		虚拟网络		44.15MB	273.69MB	317.84 MB
☐	10.244.0.21	TCP 36578	nginx-ingress-controller-68cff6666f-xcw22	虚拟网络		94.95MB	0.00B	94.95MB
☐	10.244.0.15	TCP 35202	coredns-576cbf47c7-s4mkd	虚拟网络		47.38MB	0.00B	47.38MB
☐	10.244.0.14	TCP 57324	coredns-576cbf47c7-p2dhc	虚拟网络		47.38MB	0.00B	47.38MB
☐	10.244.0.21	TCP 10254	nginx-ingress-controller-68cff6666f-xcw22	虚拟网络		7.90MB	8.75MB	16.65MB
☐	10.244.0.15	TCP 8080(http-alt)	coredns-576cbf47c7-s4mkd	虚拟网络		4.07MB	4.06MB	8.13MB
☐	10.244.0.14	TCP 8080(http-alt)	coredns-576cbf47c7-p2dhc	虚拟网络		4.00MB	4.07MB	8.07MB

每页行数: 10 1-7 共 7 条 < 1 / 1 >

- 也可以在服务拓扑中快速创建新的服务依赖



功能规格

规格

项目	规格	修改限制
业务上限	1000	配置文件 /etc/talker.yaml 中 business_max，需要重启 talker
业务内资源组上限	64	配置文件 /etc/acl-controller.yaml 中 rg_max_per_business，需要重启 acl-controller

授权

业务画像为基础功能，无需额外授权。

限制

无。

系统

管理 DeepFlow 平台自身的组件、配置、软件授权、账号、日志等。

功能使用

控制器操作

查看控制器列表信息： - 状态：正常/异常两种状态 - 监控是否有控制器失联，如果失联则认为异常。 - 角色：显示控制器的主、备、从角色 - 主/备控制器：用于提供 Web 页面访问和 API 调用入口 - 从控制器：用于向采集器同步策略 - 体系架构、操作系统、内核版本、总 CPU(核)、总内存：控制器运行环境的系统信息。 - 集群管理：表示该节点上是否安装了集群协调软件 - 目前 DeepFlow 使用 ZooKeeper 实现集群协调，用于实现控制器高可用自动切换功能。 - 当使用三台服务器部署时，可指定主区域中的三台服务器用于集群协调，此时控制器可实现主备自动切换，主控制器故障后备控制器自动切换为主。 - 当使用两台服务器部署时，若部署了多个控制器，此时仅支持手动主备切换，该部署场景下页面列表中备机的操作列会增加切换为主按钮。当主备均健康时可通过该按钮切换，当主控制器异常导致该页面无法访问时需 SSH 登录备控制器通过命令 `mt saltstack.ha_ensure_master` 实现主备切换。 - 与主控制器通信 - 当控制器集群被 NAT 网关隔离时，此项信息显示该控制器是否会使用 NAT IP 与主控制器通信 - 关联采集器数量：点击后跳转至采集器列表页面，筛选过滤显示关联采集器 - 操作： - 配置导入/导出操作：主要用于备份恢复场景。 - 配置最大关联采集器数量：当采集器数量很大且控制器缺少 N+1 冗余时，避免控制器故障时对采集器进行切换，同时如果控制器资源规格不一致时，也可限制低资源控制器管理更少的采集器 - 配置 NAT IP：当从区域的控制器和数据节点通过 NAT IP 访问主备控制器时，可设置正确的 NAT IP 使得集群之间能正常访问，另外如果部分采集器需要通过 NAT IP 访问控制器也需要进行此项设置 - 设置为备：请选择主区域中的一个从控制器为备控制器，使得主备可以自动切换 - 设置运维：当 License 不够时多余的控制器会自动设置为运维，需要补齐 License 后才能取消运维 - 对于从控制器：支持设置其关联的区域和可用区，可用区可选择一个或多个，当可用区为空时表示为区域内所有可用区的采集器提供服务。 - 点击控制器名称，可进入详情页，展现控制器 CPU、内存、磁盘、系统负载等各项监控数据。

警告： 控制器配置导入仅用于备份恢复，导出后的配置不能进行修改，且不能导入到其他控制器上。

警告： 新录入的从控制器（即通过命令行录入的 analyzer）需要在页面上与可用区关联，否则对应可用区的采集器将不会请求该从控制器，也不会向对应的数据节点发送监控数据。

另外，通过界面可直接修改控制器的主要配置： - NTP 服务器：需要设置为北京时间所在的时间同步服务器，产品页面上的所有显示时间都是以北京时间为基础。支持配置最多 4 个 NTP 服务器。另外，页面上支持查看控制器当前时间与访问者计算机的本地时间，可对比确定控制器时间是否准确。如果相差时长比较大，查看流量统计时可能产生时间错位，此时需要校准控制器或访问者计算机的本地时间。 - 云平台同步频率：需要根据云平台资源数量的大小设置合适的云平台同步频率，同步过快会给系统造成压力，同步过慢会导致系统的云平台数据与实际云平台数据差异过大，影响正常数据统计，展示等。通过查看云平台历史同步时间，能更优的设置云平台同步频率，建议云平台同步频率大于历史云平台同步平均时长。 - 上传 LOGO：支持上传图片替换页面 LOGO。 - LDAP 服务器：DeepFlow 可对接 OpenLDAP 和 Windows 的 Active Directory 域，支持 LDAP 账号以管理员的身份登录，需要填写的信息如下： - LDAP 类型：目前仅支持 OpenLDAP - 服务器地址：支持输入 IP 或域名 - 服务端口：LDAP 服务的端口号 - 是否启用 SSL：可选择是或否 - 管理员专有名称（DN）：请向 LDAP 管理员询问（格式：uid=auth,ou=people,dc=yunshan,dc=net） - 管理员 DN 密码：请向 LDAP 管理员询问 - 用户的 LDAP 搜索基准：请向 LDAP 管理员询问 - 是否同步用户信息至 DeepFlow：若选择同步，将会在 DeepFlow 中同步创建账户 - RADIUS 服务器：DeepFlow 可对接 RADIUS 服务，支持 RADIUS 账号以管理员的身份登录，需要填写的信息如下： - RADIUS 类型：目前仅支持 FreeRADIUS - 服务器地址：支持输入 IP 或域名 - 服务端口：RADIUS 服务的认证端口号 - 认证密钥：请向 RADIUS 管理员询问 - 是否同步用户信息至 DeepFlow：若选择同步，将会在 DeepFlow 中同步创建账户

建议： 输入 LDAP 服务信息以后 DeepFlow 控制器会进行简单的服务可用性检查，若检查失败将阻止配置。更改或关闭 LDAP 服务器，将会使得已经同步到 DeepFlow 中的账户无法登录，管理员可选择手动清理无效账户。

LOGO 上传 - 大小 LOGO 用于适配不同菜单最上方显示的 DeepFlow 商标 - 可上传 42K 以内的 png/jpeg/jpg/svg 图片，大 LOGO，建议尺寸为 12040 像素；小 LOGO，建议尺寸为 2424 像素 - 需点击下方的上传按钮后（而非选择完本地文件中的图片后），才可实现图片的上传

采集器操作

查看采集器统计信息

根据总流量数据能清楚知道过去一天采集和分发的流量情况，根据曲线图清楚过去一天流量走势，根据谷值、峰值直观了解最近一天最小流量值与最大流量值： - 最近一天采集流量：统计的为最近一天采集器从网卡捕获的流量 - 最近一天分发流量：统计的为最近一天采集器成功匹配分发策略，然后封装隧道报文头后的隧道流量 - 最近一天过滤流量占比：展示的为最近一天未分发流量的比例，由于分发流量需要增加隧道封装，该值可能为负数。计算方法为 $((\text{采集总流量} - \text{分发总流量}) / \text{采集总流量}) * 100\%$ 。 - 峰值/谷值：统计的为最近一天流量曲线中的最高点和最低点

通过点击最近一天采集分发流量趋势曲线图，进入二级页面查看指定时间的采集流量、分发流量、过滤流量占比。可通过时间范围筛选需要查看时间段的数据历史曲线，也可自定义调整查看的时间粒度、调整自动刷新开关。

根据 TOP 5 采集器的 CPU、内存、运行环境负载情况，能清楚知道采集器对资源的消耗，可重点关注这些采集器所在运行环境（宿主机/云服务器）是否还能支撑采集器正常工作：- TOP 5 采集器 CPU/内存用量：最近一分钟所有采集器进程的内存/CPU 用量 TOP 5。- TOP 5 运行环境负载：最近一分钟所有采集器所在运行环境的系统负载，注意不支持采集器运行于 Windows 操作系统中的场景。- 通过点击 TOP 5 表格右上角的扩大图标能查看对应数据的 TOP 100 情况：- 通过点击 TOP 5 表格右上角的波浪图标能查看 TOP 5 对象的历史统计数据。

根据 TOP 5 采集器采集流量能知道采集器所在的宿主机/云服务器流量活跃情况，根据分发流量能知道采集器的分发流量情况，可重点关注是否消耗过多资源：- Top5 采集器一小时采集/分发 bps：采集器采集/分发的最近一小时的流量速率 - Top5 采集器一小时采集/分发 pps：采集器采集/分发的最近一小时的包量速率 - Top5 云服务器一小时采集/分发 bps：以云服务器虚拟接口为粒度的统计量 - Top5 云服务器一小时采集/分发 pps：以云服务器虚拟接口为粒度的统计量 - Top5 采集器一小时采集丢包数：统计的为采集器在采集流量时，因资源瓶颈发生的丢包数的总和。根据采集器丢包数，能快速确定平台丢包的采集器，从而去定位采集器所在的宿主机/云服务器定位为什么会出现丢包，同时在数据不准确时能确实是否因丢包引起。- 通过点击 TOP 5 表格右上角的扩大图标能查看对应数据的 TOP 100 情况：- 通过点击 TOP 5 表格右上角的波浪图标能查看 TOP 5 对象的历史统计数据。

[查看采集器列表](#)

根据采集器列表，清楚知道采集器安装部署以及运行情况：- 名称：采集器的名字 - KVM：资源管理中显示的宿主机名称，通常是宿主机的 hostname - 容器-V/容器-P：资源管理中显示的容器节点名称，通常是容器节点的 hostname - ESXi：资源管理中显示的宿主机名称 - 专属服务器：服务器的 hostname - Workload-V：资源管理中显示的云服务器名称和云服务器 ID 后缀（用于避免重复） - Workload-P：资源管理中显示的裸金属类型的云服务器的名称 - 区域、可用区：采集器所在的区域和可用区 - 专属服务器，为对应控制器的区域和第一个可用区 - 其他类型的采集器，等于其运行环境所在的区域和可用区 - 组：采集器所属的组，点击可跳转至组页面，未指定组的采集器属于默认组。- 类型：说明采集器的运行环境 - KVM：采集器 Trident 进程运行于宿主机（如 KVM）上 - 容器-V/容器-P：采集器 Trident 以 DaemonSet 的方式运行于每个容器节点（K8S Node）上 - ESXi：采集器 Trident 进程运行于 vSphere ESXi 上的专用虚拟机中，采集 ESXi 上所有业务虚拟机的镜像流量 - 专属服务器：采集器 Trident 进程运行于专属服务器上，采集物理交换机镜像流量 - Workload-V：采集器 Trident 进程运行于业务虚拟机内部 - Workload-P：采集器 Trident 进程运行于业务裸金属服务器内部 - 隧道解封装：采集器 Rosen 进程运行于独立服务器上，用于解除分发流量的隧道封装 - 运行环境 IP：显示采集器运行环境的 IP 地址 - 专属服务器此项为空 - ESXi 类型的采集器，此处显示为 ESXi 的

IP 地址 - 点击运行环境的地址，可跳转至资源管理页面或数据节点列表页面 - 控制 IP：为采集器与控制器通信的 IP 地址。 - 控制 MAC：为采集器与控制器通信的 MAC 地址。 - 状态/异常：展示采集器目前状态，包括未注册/运行/失联/禁用。当采集器运行过程中存在异常时该列会显示红色感叹号，目前支持的异常信息包括 - 自检失败：日志所在磁盘剩余空间不足 100MB - 自检失败：可用内存不足 - 分发熔断触发 - 分发流量达到限速 - 到分发点的网关 ARP 无法找到 - 到数据节点的网关 ARP 无法找到 - 软件版本/完整软件版本：显示 Trident 软件的版本号，用于故障排查和升级指示 - 启动时间：表示的为采集器进程启动的时间。 - 体系架构、操作系统、内核版本、总 CPU(核)、总内存：采集器运行环境的系统信息。 - 控制器：表示采集器请求策略的从控制器 IP 地址（同时也是发送监控信息的目的数据节点 IP 地址），点击可跳转至控制器列表中。 - 当前访问控制器：当控制器在 SLB 后为采集器提供服务时，此字段展示采集器正在请求的真实控制器 IP - 控制器同步时间：展示的采集器最近一次与控制器同步云平台信息的时间。 - 数据节点：表示采集器发送数据的目标数据节点 - 当前访问数据节点：当数据节点在 SLB 后未采集器提供服务时，此字段展示采集器正在请求的真实数据节点 IP - 数据节点通信时间：表示采集器与数据节点的最后通信时间，注意此项值的更新会存在一些 Cache 导致的滞后 - 操作：支持注册、启用/禁用、删除、查看虚拟接口实时流量 - 注册：对新发现的采集器进程认证，控制器注册之前的状态为未注册，不会执行任何动作。注意此操作不可逆，除非删除采集器。 - 启用/禁用：对采集器启用或禁用。 - 删除：删除无用的采集器。 - 查看虚拟接口实时流量：见下节。

此外，列表顶端的按钮提供了对采集器进行批量或全量操作的能力： - 启用：多选采集器，批量执行启用操作 - 禁用：多选采集器，批量执行禁用操作 - 注册：多选采集器，批量执行注册操作 - 加入采集器组：多选采集器，批量执行加入采集器组操作 - 关联控制器全局均衡：对所有采集器执行全量的均衡操作，使得各可用区内从控制器对应的采集器数量均衡。注意：被切换的采集器进程将会自动重启，建议仅当显著不均衡时执行此操作。当无需均衡时，弹出页面的确定按钮不可点击。 - 新增隧道解封装采集器：输入名称、所属采集器组、控制 IP、区域、可用区 - 控制 IP：填写控制器可连通的 IP 地址

点击某采集器名称，可查看采集器详情、监控数据、运行日志： - 基本信息：展示目前采集器基本信息、采集器的环境配置、状态信息及采集器所处运行环境的信息。 - 配置信息：展示采集器对应的采集器组配置的详细信息。 - 监控数据：展示目前采集器详情页的所有监控图表。 - 运行日志：支持多区域部署场景下，查看非主区域采集器的运行日志。展示记录与 ES 中的采集器的所有日志，且 WARN 日志标黄、ERR 日志标红。

通过查看某采集器关联的具体云服务器接口的采集、分发流量统计数据，清楚了解目前活跃的云服务器或容器 POD 接口及对应的分发策略： - 采集总流量/采集总包量/分发总流量/分发总包量：统计的都是查询的最近 1 小时的对应统计数据。 - 通过点击列表最右端的分发策略按钮，查看具体云服务器接口对应的分发策略，可快速知道云服务器发送流量分发时匹配的具体分发策略，展示采集点过滤规则可匹配

的分发策略。 - 专属服务器的采集器没有关联的云服务器，无法查看此信息。容器-V、容器-P 类型的采集器展示的信息为容器节点上的 POD 的接口，且无法查看关联的分发策略。

查看采集网卡列表

将采集器与控制器同步的网卡展示，方便管理 - 名称：网卡名称，如容器 POD 网卡/虚拟机内部网卡/宿主机物理网卡/容器节点自身网卡等 - MAC: 上述网卡对应的 MAC 地址 - 资源类型：上述网卡对应的资源类型 - 资源名称：可跳转到对应的资源列表 - 采集器：上述网卡对应的采集器，支持跳转到采集器列表 - 采集器类型：支持筛选过滤，目前只支持 KVM、容器-V/P、Workload-V/P 采集器类型 - 采集网卡：上述网卡对应的采集网卡，例如容器 POD 内看到的网卡 eth0，在容器节点采集时对应的为 veth-xxx - 采集 MAC: 采集网卡对应的 MAC - 采集器运行环境：采集器支持跳转到资源列表

采集器组及配置

通过将相同类型的宿主机/云服务器归纳为一组，方便统一管理： - 默认：如用户未给采集器自定义组，则都划分为默认组中。默认组不可能修改和删除。 - 采集器以最后加入的组为准。

通过设置采集器默认组配置，限制默认采集器组中每个采集器的资源使用上限： - 全局配置：输入框为空时代表的系统初始值，将一个有值的框置空意味着还原为系统默认。高亮显示相对于系统初始值有变化的参数值 只能修改不能删除。 - 其他组配置：参数输入框为空时，表示使用采集器组的全局配置，高亮显示相对于全局配置有变化的参数值。 - 资源限制 - CPU 限制：限制采集器使用宿主机/云服务器上的逻辑 CPU 核。例如设置为 2，则实际采集器最大消耗为 200%。 - 内存限制：限制采集器使用宿主机/云服务器上的内存。当运行环境内存余量小于该值时，采集器进程启动、启用时自检会失败，并会在页面上显示自检失败信息。 - 采集包速率：采集包速率主要限制采集器对整体资源的限制，需要通过在用户环境综合 CPU，内存给出综合的合适的限制。 - 分发流限制：分发流量占用的是用户物理出带宽，可根据用户物理出带宽分配合理设置分发可使用的量。 - 分发熔断阈值：当分发接口出方向流量达到或超过阈值时采集器停止分发，当连续 5 个监控周期的流量低于(阈值-分发流量限制)*90%时恢复分发。注意：输入 0 表示关闭此功能，若设置该值请确保必须大于分发流限速的两倍。 - 分发熔断监控间隔：分发接口出方向流量平均速率的监控间隔。 - 日志发送速率：最大日志发送速率，单位为条/秒。 - 日志打印等级：定义打印的日志级别。 - 线程数限制：限制采集器运行环境中 trident 进程内线程数量。 - 进程数限制：限制采集器运行环境中 trident 进程启动的其他子进程数量。 - 基础配置参数 - 采集网口：Trident 采集流量的网口正则表达式，注意当需要匹配以 vtap 开头的网口时需要在头部增加^，例如^vtap - 流量过滤：采集器采集流量时的过滤规则，默认全采集，支持 BPF 语法输入规则 - 采集包长：采集器采集包的最大长度，若超过此长度，则截断该包。 - 流量采集方式：Linux 环境中的流量采集方式，默认为自适应，可选项为自适应、AF_PACKET V2、

AF_PACKET V3 - 解封装隧道类型：无、VXLAN、VXLAN-VXLAN、TCE-GRE、IPIP、VXLAN-IPIP - 虚拟机 MAC 解析：在 KVM 上一般而言 vtap 口的 MAC 地址后 5 字节与虚拟机内部的 MAC 地址一致（即通过接口 MAC 进行关联），但部分云平台例如 Contrail 是通过接口的名称进行关联，此时需要将该项配置修改为接口名称 - 虚拟机 XML 文件夹：默认为/etc/libvirt/qemu/，当虚拟机 MAC 解析为虚拟机 XML 时，采集器需检查此项配置，若无法找到对应的文件夹时需要拒绝此项配置并打印 warn 日志 - 最长同步间隔：当云平台信息变更时，分发策略同步最长时间。特指采集器与控制器同步云平台信息的时间，比如子网内新增一个云服务器，分发时针对子网进行分发那么这台新增的云服务器也需要分发。 - 最长逃逸时间：控制器在失联时采集器还能继续使用控制器最后下发的信息进行工作，但采集器会在逃逸时间到达之后自动进入禁用状态，默认值域为[600,2592000] - UDP 最大 MTU：当使用裸 UDP 套接字传输数据时，如果采集器发送网卡的 MTU 不是沿途上的最小 MTU，需要在此进行配置。另外需要注意的是公有云上需要调低 MTU 至 1400 避免公有云篡改 UDP 大包的尾部字符。 - 裸 UDP 外层 VLAN：当使用裸 UDP 套接字传输数据时，如果需要添加 VLAN 头发包可填写此值 - 是否请求 NAT IP：当采集器需要通过 NAT IP 请求关联的控制器或数据节点时填写此项为是 - 日志存储时长(天)：默认 30，必填，可填写 7-365 之间的任意整数 - 全景图配置参数 - 数据套接字：采集器发送指标量数据和流日志数据使用的套接字，默认为 UDP，可选择 TCP 提升传输可靠性（但是会占用更多的带宽资源） - PCAP 套接字：采集器发送 PCAP 数据使用的套接字，默认为 UDP，可选择裸 UDP 提升性能，也可选择 TCP 提升传输可靠性（但是会占用更多的带宽资源） - HTTP 日志代理客户端：默认为 X-Forwarded-For，为空时 HTTP 日志不提取代理客户端 IP 字段，可自定义 Header - HTTP 日志 TraceID：可选项分别表示 Zipkin、Jaeger、Skywalking 使用的 HTTP Header，可自定义 Header - 应用日志解析包长：采集 HTTP、DNS 日志时的解析的包长 - 流日志采集速率(每秒)：每秒采集的流日志条数，超过时采样，默认值域为[100,1000000] - 应用日志采集速率(每秒)：每秒采集的 HTTP 和 DNS 日志条数，超过时采样，默认值域为[100,1000000] - 包分发配置参数 - 分发套接字：采集器分发流量时使用的套接字，默认为裸 UDP，可选择 UDP 提升适配性（部分场景下使用裸 UDP 无法正确获取网关 MAC 地址） - 内层附加头：分发流量中是否添加内层 802.1Q 附加头，用于向第三方分析工具发送流量标签 - 基础功能开关 - 同步资源信息：是否打开资源信息同步，用于在没有云平台资源信息 API 同步能力的场景下通过采集器同步 KVM 上的虚拟机、虚拟网卡、MAC、IP 等信息 - 日志发送：开启发送功能，是会将采集器产生的日志都发送至控制器的 /var/log/trident/\$采集器.log 目录下。如某台采集器出现异常时，又不方便登录采集器时，可直接将采集器的日志发送到控制器，在控制器集中查看定位原因 - 时钟同步：默认开启，仅用于采集器进程内部时钟计算，开启后不会影响采集器所在操作系统的时钟 - 全景图功能开关 - 指标数据：采集器是否发送用于全景图的指标数据 - 非活跃端口指标数据 - 网络性能指标数据：关闭时，采集器不计算 TCP、UDP 性能指标数据，仅计算最基本的指标网络层吞吐 - 应用性能指标数据：采集器是否计算应用层指标数据 - 秒粒度指标数据：采集器是否发送全景图中的秒粒度流统计数据 - 流日志数据：采集器是否发送全景图中的流日志数据 - 流日志开启采集

点：当不希望采集器发送所有采集点的流日志数据时，可配置采集点过滤列表仅发送对应采集点的流日志数据，一般用于降低流日志发送压力；默认为全部，表示所有采集点；为空时表示关闭流日志采集；- 应用日志开启采集点：逻辑与过滤流日志类似，默认为全部 - 包分发功能开关 - 全局去重：当不希望采集器进行全局去重时，可关闭此开关

注意：- 专属服务器类型的采集器可加入组，但仅能使用组的启用、禁用操作，资源配置对此类采集器不生效。- KVM、Workload-V Linux 类型采集器会根据性能动态调整采集包长（设置截断长度）

通过对指定采集器组中每个采集器组的资源使用上限进行配置，可根据宿主机/云服务器不同资源可用情况给采集器组做灵活的资源配置：- 采集器组：指定采集器组配置优先级高于默认配置。- 新增组配置：用户只需填写和默认配置的差异化部分，其余未修改部分都与默认配置保持一致。- 删除：默认配置不支持删除。

数据节点操作

数据节点列表

查看数据节点列表信息：- 区域、可用区：数据节点关联的区域、可用区列表。数据节点会接收关联可用区中采集器的遥测数据、压缩包头和原始流信息并写入数据库或磁盘中。从控制器所在服务器一定是数据节点，通过设置从控制器关联的区域、可用区即可实现对数据节点的设置。- 状态：正常/异常两种状态，主要监控是否有数据节点失联，如果失联则认为异常 - 关联采集器数量：目前关联的采集器数量 - 最大关联采集器数量：该值用于采集器切换数据节点时的超载保护。支持修改，输入值小于当前关联的采集器数量时，将会触发超出的采集器切换，被切换的采集器进程将会自动重启。数据节点所在服务器上也会运行采集器，即最大关联数量至少为 1。- 体系架构、操作系统、内核版本、总 CPU(核)、总内存：数据节点所处运行环境的系统信息。- 点击数据节点名称，可进入详情页，展现数据节点 CPU、内存、磁盘、系统负载、队列丢包等各项监控数据。- 操作 - 设置运维：当 License 不够时多余的数据节点会自动设置为运维，需要补齐 License 后才能取消运维 - 配置区域和可用区：支持设置其关联的区域和可用区，可用区可选择一个或多个，当可用区为空时表示为区域内所有可用区的采集器提供服务。- 配置最大关联采集器数量：当采集器数量很大且数据节点缺少冗余时，避免数据节点故障时对采集器进行切换，同时如果数据节点资源规格不一致时，也可限制低资源数据节点管理更少的采集器 - 配置 NAT IP：如果部分采集器需要通过 NAT IP 访问数据节点，需要进行此项设置 - 配置参与聚合：当同一个区域内数据节点无法全部互相通信时，可以将其中某个可用区的数据节点设置为参与聚合，其他可用区仅需要和此可用区内的数据节点通信而无需相互通信

存储配置

全景图统计数据分为流统计、流量日志两种数据库，分钟粒度数据默认保存 1 周、秒粒度数据默认保存一天，过期数据自动删除。用户可基于已有的数据源自定义新

的数据源，最多支持 10 个数据源。点击每个数据源，可查看数据的磁盘占用总量、每个时间间隔的占用增量、所在磁盘的剩余空间，用户可据此设定合适的保留时间。

网络诊断中 PCAP 数据默认保存 1 周，1 周以前的数据会自动删除。该页面展现统计数据的磁盘占用总量、每个时间间隔的占用增量、所在磁盘的剩余空间，用户可据此设定合适的保留时间。

系统中的监控数据默认保存 1 周，1 周以前的数据会自动删除。该页面展现监控数据的磁盘占用总量、每个时间间隔的占用增量、所在磁盘的剩余空间，用户可据此设定合适的保留时间。

账号管理

账号支持多个管理员、多个租户，管理员之间权限一致，租户可被管理员授予不同资源的查看权限。

账号源：目前有 DeepFlow、LDAP、SSO、RADIUS、OAuth、腾讯专有云（TCE）- 腾讯专有云中，对接账号为运营端账号，如下：CVM 运维人员、VPC 运维人员、COS 运维人员、DB 运维人员、平台运营人员，以普通管理员类型展示在管理员列表

用户状态分为：启用、禁用、冻结。

- 管理员列表
 - 超级管理员可以增删改查所有管理员账号及租户账号
 - 非超级管理员只可修改自己的账号
- 租户列表
 - 管理员可以增删改查授权所有租户账号
 - 租户只可修改自己的账号
 - 租户可被授权的资源有：
 - VPC，及 VPC 相关的云服务器、VPC、子网、路由器、DHCP 网关、IP 地址、NAT 网关、负载均衡器、云数据库 RDS、云数据库 REDIS
 - 容器命名空间，及命名空间关联的 Ingress、服务、工作负载、ReplicaSet、POD
 - 租户可看到的目录有：
 - 全景图（流量搜索、流量下载）、视图（视图列表）、告警（告警策略、告警事件）、报表（报表策略、报表下载）、资源（所有页面）、系统（账号管理、操作日志）

租户支持的操作有：

资源	资源权限	操作权限-支持	操作权限-不支持

资源池->区域	全部可见	导出 CSV	新建区域、编辑、删除
资源池->可用区	全部可见	导出 CSV	新建可用区、编辑、删除
资源池->云平台	全部可见	导出 CSV	新建云平台、配置同步间隔、同步时间统计、编辑、删除、启用/禁用、录入容器集群、附属容器集群参数 不可点击跳转
计算资源->云服务器	关联 VPC、命名空间下资源	导出 CSV、详情-网卡列表	新建云服务器、编辑、删除
计算资源->云服务器->网卡详情	关联 VPC 下资源	导出 CSV	新建、编辑、删除
网络资源->VPC	关联 VPC、命名空间	导出 CSV	新建 VPC、编辑、删除
网络资源->子网	关联 VPC、命名空间下资源	导出 CSV	新建子网、编辑、删除
网络资源->路由器->路由器	关联 VPC 下资源	导出 CSV，查看路由规则、路由规则导出 CSV	-
网络资源->路由器->路由器规则	关联 VPC 下资源	导出 CSV	-
网络资源->DHCP 网关	关联 VPC 下资源	导出 CSV	-
网络资源->IP 地址	关联 VPC、命名空间下资源	导出 CSV	-
网络服务->安全组->安全组	全部可见	导出 CSV，查看规则、规则支持导出 CSV	-
网络服务->安全组->安全组规则	全部可见	导出 CSV	-
网络服务->NAT 网关->NAT 网关	关联 VPC 下资源	导出 CSV，查看规则、规则支持导出	-

		CSV	
网络服务->NAT 网关->NAT 网关规则	关联 VPC 下资源	导出 CSV	-
网络服务->负载均衡器->负载均衡	关联 VPC 下资源	导出 CSV，查看规则、规则支持导出 CSV	-
网络服务->负载均衡器->负载均衡规则	关联 VPC 下资源	导出 CSV	-
网络服务->对等连接	全部可见	导出 CSV	新建对等连接、编辑、删除
网络服务->云企业网	全部可见	导出 CSV	-
存储服务->云数据库 RDS	关联 VPC 下资源	导出 CSV	-
存储服务->云数据库 REDIS	关联 VPC 下资源	导出 CSV	-
容器资源->容器集群	全部可见	导出 CSV	-
容器资源->命名空间	关联 VPC、命名空间下资源	导出 CSV	编辑
容器资源->容器节点	全部可见	导出 CSV	编辑
容器资源->INGRESS->INGRESS	关联 VPC、命名空间下资源	导出 CSV、查看转发规则、规则支持导出 CSV	-
容器资源->INGRESS->转发规则	关联 VPC、命名空间下资源	导出 CSV	-
容器资源->容器服务->容器服务	关联 VPC、命名空间下资源	导出 CSV，查看端口映射、端口映射支持导出 CSV	编辑

容器资源->容器服务->端口映射	关联 VPC、命名空间下资源	导出 CSV，查看端口映射、端口映射支持导出 CSV	编辑
容器资源->工作负载	关联 VPC、命名空间下资源	导出 CSV	编辑
容器资源->ReplicaSet	关联 VPC、命名空间下资源	导出 CSV	编辑
容器资源->容器 POD	关联 VPC、命名空间下资源	导出 CSV	编辑
业务画像->业务列表	租户自己创建业务	新建业务、批量删除、编辑、删除	-
业务画像->点击某业务进入业务详情页	租户自己创建业务	新建资源组、批量删除、编辑、删除；支持新建服务依赖、批量删除、编辑、删除	-
业务画像->服务拓扑	租户自己创建业务	新建服务依赖、保存	-
其他资源->物理网元	全部可见	导出 CSV	新建物理网元、编辑、删除
其他资源->采集点	全部可见	导出 CSV	新建采集点、编辑、删除
其他资源->物理链路	全部可见	查看拓扑、导出 CSV	新建链路、重置拓扑、编辑、删除
其他资源->图例	全部可见	-	新建图例、编辑、删除、恢复默认

操作日志

呈现系统用户的页面操作日志用于审计。注意：查看操作日志允许选择的最大时间间隔为 1 个月。

授权管理

选择软件授权模式时，DeepFlow 部署后需要上传授权文件才能正常使用。当授权文件缺失或错误时的页面行为是：管理员登录后只能访问授权管理页面，租户无法登录。申请授权文件时，需要提供如下信息：- 客户名称 - 授权功能 - 数据节点数量 - 控制器数量、是否为控制器集群 - 不同类型的采集器数量：KVM、ESXi、Hyper-V、容器-V、容器-P、专属服务器、隧道解封装、Workload-V、Workload-P - A 类授权可授予一个宿主机上的所有虚拟化流量采集，或者一个容器上的所有 POD 流量采集 - B 类授权可授予公有云或者类公有云部署场景 - C 类授权可授予一个宿主机上的所有虚拟化、容器流量采集 - 授权时间：目前版本支持为 POC 签发 30 日授权，为商用客户签发永久授权，POC 客户过期需重新申请。- 授权管理页面中显示的控制器控制 IP 地址、控制器主机名信息，需要与授权文件中的对应信息匹配。

当授权过期时，页面及 API 会增加一定概率的延时，且时长随过期时间增大同步增大，需要用户及时更新授权文件。

用户可上传多个授权文件，授权到期时间以最近过期的一个授权文件为准，控制器、数据节点、采集器授权数量等于所有授权文件的加和。仅当存在一个支持控制器集群的授权文件时才支持部署大于 1 个控制器。

费用中心

选择按量计费模式时，DeepFlow 部署后需在费用中心使用代金券充值，充值后将对云服务器、容器节点提供的流量监控服务按天计费。

代金券充值需提供如下信息：- 客户名称 - 合同编号 - CRM_ID - DeepFlow 版本 - 控制器主机名：任意控制器主机名都可 - 控制器 IP 地址：任意控制器主机名都可

计费规则如下：- 当天只要启用过云服务器、容器节点对应的采集器，就会发生计费 - 金额不足导致扣款失败时，则直接禁用云服务器、容器节点对应的采集器

通知信息如下：- 每个月 1 号时，会发送上个月消费的月账单邮件 - 第一次充值代金券时，会发送正式启动邮件 - 后续每次充值代金券时，会发送充值通知邮件 - 预估可用天数不足 30 天，将每天发送告警邮件 - 无可用天数时，会发送停服告警邮件

技术支持

下载系统进程日志用于技术协助，从 Web 页面登录控制器控制台用于故障排查。注意：下载系统进程日志允许选择的最大时间间隔为 1 周。

提供界面上的 ssh 工具：webssh，该服务支持 rz、sz 命令，只开放给超级管理员。

提供上传下载文件服务，默认上传文件保存地址在 /var/www/lcweb/web-tools，配置路径在系统->配置管理，配置参数名是 file_storage，默认只开放给超级管理

员。详细参数设置参考部署扩容->控制器与数据节点部署->页面定制配置中的 web 安全配置说明。

典型场景

在多区域中部署 DeepFlow

当 DeepFlow 部署在多区域环境中时，需要确保每个区域都部署数据节点和从控制器，并在控制器页面将其与所在的区域关联。

若一个区域的采集器无法访问同区域另一个可用区中的从控制器，从控制器不能与区域中所有可用区关联，必须进行分可用区的关联。

功能规格

规格

项目	规格	修改限制
全景图分钟级数据保留时长	7 天	数据节点-全景图存储配置页面
全景图秒级数据保留时长	1 天	数据节点-全景图存储配置页面
流回溯 PCAP 文件保留时长	7 天	数据节点-PCAP 存储配置页面
系统监控数据保留时长	7 天	数据节点-系统数据存储配置页面

授权

系统管理为基础功能，无需额外授权。

限制

- LDAP：如果需要使用 SSL 来校验，LDAP 服务器必须关闭客户端校验，否则通过 SSL 校验会失败
- 系统管理页面展现的所有统计数据，时间间隔最小支持 1 分钟
- 控制器未完成采集器注册时，在采集器统计信息中能看到未注册采集器的 CPU、内存消耗信息
- 删除采集器重新注册后，从该采集器已存在的系统告警事件信息详情中不能正常跳转到采集器详情页
- 纯 IPv6 环境支持限制
 - 采集器暂不支持解析 IPv6 的 VXLAN、ERSPAN 隧道
 - 采集器在 Windows 操作系统中不支持运行于纯 IPv6 环境下
 - 目前的 Kibana 版本不支持在纯 IPv6 下使用
- 流量采集接口后端 MAC 地址获取方式：
 - KVM 环境获取虚拟机 MAC（需要在系统-采集器-配置中指定）
 - 接口 MAC（默认）：通过 tap 口的 MAC 后 5 字节确定虚拟机 MAC，第一字节忽略

- 接口名称：Contrail 和青云采用此方式，tap 口的名称后缀代表 MAC 地址的后 5 字节，同样第一字节忽略
- 虚拟机 XML：通过 QEMU 的虚拟机 XML 文件提取网卡 MAC，默认 XML 文件夹路径为/etc/libvirt/qemu/
- K8s 环境获取 Pod MAC（无需配置）
 - 接口 MAC（默认）：通过 Pod Namespace 内网卡 MAC 地址直接获得
 - ARP 流量（自动切换）：当 Trident 所在 Pod 无法被赋予 SYS_ADMIN、SYS_PTRACE 或 HostPID 权限时自动切换为此方式

规范

控制器高可用

- HA 依赖组件
 - zookeeper
 - hamgrd
- zookeeper
 - 机制说明
 - 集群节点间使用 TCP 协议（2888 和 3888 端口）进行探测，默认探测间隔 2s
 - follower 与 leader 之间初始化连接时能容忍的超时时间是 10s，follower 与 leader 之间同步状态时能容忍的超时时间是 4s
 - 配置（/var/lib/zookeeper/conf/zoo.cfg）

```
# 心跳时间间隔，默认 2s
tickTime=2000
# 存储目录，记录集群 id, pid 等内容
dataDir=/var/lib/zookeeper/data/
# 客户端端口，hamgrd 使用该端口连接 zk
clientPort=2181
# follower 与 leader 之间初始化连接时能容忍的最多心跳数
initLimit=5
# follower 与 leader 之间同步状态时能容忍的最多心跳数
syncLimit=2
# 集群服务器列表
server.{ host_index + 1 }={{ host }}:2888:3888
```

- hamgrd
 - 机制说明
 - 访问 zookeeper 的 2181 端口，默认最大超时时间为 10s，如果连不上则认为需要做切换
 - 切换时间大约 2min，切换动作如下：

- 启停网卡
- 刷新主控制器配置文件，并启动主控制器上相关进程
- 停止备控制器上的相关进程
- 配置（暂无配置文件， /usr/local/deepflow/hamgrd/hamgrd.py）


```
daemon = KazooMonitor(
    KazooClient(auth_data=[("digest", f"{config.zk_user}:{con
fig.zk_pass}")]),
    connection_retry=KazooRetry(max_tries=5,
                                delay=2,
                                max_delay=10)))
```

二次开发

DeepFlow 可输出的三类数据，用户可基于 DeepFlow 平台进行二次开发。

- 全景图：通过调用控制器 Restful API 获取指标数据
 - 吞吐量
 - TCP 时延
 - TCP 传输
 - TCP 异常
- 流日志：通过请求数据节点 ZeroMQ 队列订阅实时流数据
 - L4 层七元组（五元组+源目 VPC）网流日志
- 流量分发：通过接收采集器隧道封装流量获取实时网包数据
 - VXLAN 封装
 - ERSPAN 封装
 - 内层 VLAN 封装

应用/网络

指标数据

数据的类型

DeepFlow 支持如下几类指标量的输出：

- 网络层吞吐
- 传输层吞吐
- 应用层吞吐
- 传输层负载
- 传输层时延
- 应用层时延

- 传输层性能
- 传输层异常
- 应用层异常
- 资源属性
- 路径属性
- 其他属性

上述指标数据结合不同的算子（目前支持 Sum、Avg、Max、Min），总共可提供一百多种指标量统计值。

数据的资源标记

DeepFlow 数据节点运行自研的分布式时序数据库，用户可以对所有指标量从多种角度进行聚合查询。得益于 DeepFlow 控制器和云平台、容器平台进行资源同步，并将流量的源、目的地址与资源进行关联，用户可从十余个维度对资源进行分组聚合查询：- 资源池：区域（Region）、可用区（AZ, Availability Zone）- 虚拟化：宿主机、云服务器 - 容器：节点、命名空间、工作负载、POD - 应用：业务、资源组（由业务画像生成）- 网络：VPC（Virtual Private Cloud）、子网、IP 地址

数据的其他属性标记

除了上述资源维度以外，用户还可通过其他流量属性字段对指标数据进行过滤或分组查询，包括：

- 采集点：对流量的采集位置进行过滤，物理网络采集点可在资源管理-其他资源-采集点中进行维护
- 监控资源角色：对监控资源的客户端、服务端角色进行过滤
- 流属性字段
 - IP 类型：对 IPv4、IPv6 流量进行过滤
 - 网络协议：对流量的 L3 网络协议类型进行过滤
 - 服务端口：对流量的 L4 服务端口进行过滤，支持 TCP 和 UDP 端口

数据的时间精度

参照系统/数据节点/全景图存储配置中的数据源列表

指标量说明

详细指标量列表参考[全景图功能规格](#)。

API 说明

API 列表

页面	功能	API URL

流统计拓扑图	获取拓扑图数据	/v1/stats/lake/topo/Distribution
流-双端统计	获取柱状图数据	/v1/stats/lake/peers/Distribution
	获取饼状图数据	/v1/stats/lake/peers/Distribution
	获取 Top 折线图数据	/v1/stats/lake/peers/Distribution
	获取表格数据	/v1/stats/lake/peers/DetailList
流-资源统计	获取柱状图数据	/v1/stats/lake/instance/Distribution
	获取饼状图数据	/v1/stats/lake/instance/Distribution
	获取 Top 折线图数据	/v1/stats/lake/instance/Distribution
	获取表格数据	/v1/stats/lake/instance/DetailList
流-资源盘点	获取表格数据	/v1/stats/lake/resource/DetailList
链路追踪	获取物理拓扑图数据	TODO
	获取表格数据	TODO
流量曲线	获取资源流量曲线数据	/v1/stats/lake/instance/History/
	获取双端流量曲线数据	/v1/stats/lake/peer/History/
	获取资源包数曲线数据	/v1/stats/lake/packet/History/
	获取双端包数曲线数据	/v1/stats/lake/packet_peers/History/

资源分组条件参数

资源分组条件描述了统计数据资源分组标准，由如下条件组成：

页面 输入 框	字段名	字段类型	取值说明
监控 资源 分组	TYPE	StringType	可选值：region, az, host, pod_node, l3_epc, subnet, vm, pod, group, ip, business, pod_group, pod_ns
关联 资源 分组	EXTERNAL_TYPE	StringType	可选值：空, region, az, host, pod_node, l3_epc, subnet, vm, pod, group, ip, business, pod_group, pod_ns
IP 掩 码长 度	IP_NETMASK	IntType	取值范围：1-128
流属	SUB_TYPE	StringType	可选值：空, ip_version, protocol,

性分			server_port, tap_type
组			
额外	NODE_KNOWLEDGES	ListType	由 StringType 组成，对应不同分组
信息	INCLUDE_OTHERS	Bool	是否返回其他
	INCLUDE_HISTORY	Bool	是否返回 Top 折线图
	INCLUDE_COMPLETION	Bool	是否补齐拓扑图

监控资源过滤条件参数

页面输入框	字段名	字段类型	取值说明
区域	REGION_IDS	ListType	由 IntType 组成
可用区	AZ_IDS	ListType	由 IntType 组成
宿主机	HOST_IDS	ListType	由 IntType 组成
云服务器	VM_IDS	ListType	由 IntType 组成
容器节点	POD_NODE_IDS	ListType	由 IntType 组成
容器 POD	POD_IDS	ListType	由 IntType 组成
业务	BUSINESS_IDS	ListType	由 IntType 组成
资源组	GROUP_IDS	ListType	由 IntType 组成
VPC	L3_EPC_IDS	ListType	由 IntType 组成
子网	SUBNET_IDS	ListType	由 IntType 组成
IP 地址	IPS	ListType	由 StringType 组成
容器组	POD_GROUP_IDS	ListType	由 IntType 组成
容器命名空间	POD_NS_IDS	ListType	由 IntType 组成

关联资源过滤条件参数

页面输入框	字段名	字段类型	取值说明
区域	REGION_IDS	ListType	由 IntType 组成
可用区	AZ_IDS	ListType	由 IntType 组成
宿主机	HOST_IDS	ListType	由 IntType 组成
云服务器	VM_IDS	ListType	由 IntType 组成
容器节点	POD_NODE_IDS	ListType	由 IntType 组成
容器 POD	POD_IDS	ListType	由 IntType 组成
业务	BUSINESS_IDS	ListType	由 IntType 组成
资源组	GROUP_IDS	ListType	由 IntType 组成

VPC	L3_EPC_IDS	ListType	由 IntType 组成
子网	SUBNET_IDS	ListType	由 IntType 组成
IP 地址	IPS	ListType	由 StringType 组成
容器组	POD_GROUP_IDS	ListType	由 IntType 组成
容器命名空间	POD_NS_IDS	ListType	由 IntType 组成

流量属性过滤条件参数

页面输

入框	字段名	字段类型	取值说明
采集点	TAP_TYPES	ListType	由 IntType 组成
IP 类型	IP_TYPE	IntType	可选值：4-ipv4, 6-ipv6
监控资源	DIRECTION	StringType	可选值：c2s-客户端, s2c-服务端
网络协议	PROTOCOLS	ListType	由 IntType 组成，对应不同协议的数值
服务端口	SERVER_PORTS	ListType	由 IntType 组成
省份	TAG_VALUES	ListType	由 StringType 组成，对应不同省份名称
播送类型	CAST_TYPE	StringType	可选值：unicast-未知单播, broadcast-广播, multicast-组播
TCP 标志位	TCP_FLAG	IntType	可选值：1-SYN, 18-SYN+ACK, 16-ACK, 24-PSH+ACK, 17-FIN+ACK, 20-RST+ACK, 255-未分类
统计位置	TAP_SIDE	StringType	可选值：c-客户端, s-服务端
端到端路径	PEER_FILTERS	ListType	由 StringType 组成，对应不同通信：include_internal-监控资源间, include_external-与关联资源, include_internet-广域网
是否仅显示容器及资源组	INCLUDE_SERVICE	IntType	可选值：1-容器及资源组, 2-所有服务
曲线图端到端路径	PEER_FILTERS_HISTORY	StringType	可选值：include_internal-监控资源间, include_external-与关联资源, include_internet-广域网

指标量过滤条件参数

页面输入框	字段名	字段类型	取值说明
指标量	METRICS	StringType	
算子	OPERATOR	StringType	可选值：sum, avg, max, min
最大值	MAX	IntType	
最小值	MIN	IntType	

流日志数据

数据格式

DeepFlow 采集器将相同七元组（源目 MAC 避免多租户 IP 冲突、源目 IP、源目端口、协议）聚合为网流，开发人员可请求 DeepFlow 数据节点，通过 Clickhouse API 读取或 ZeroMQ 实时获取所有采集器的网流数据。

流日志

区块	页面显示 字段名	存储字段名	存储类型	零值无意义	说明
链路层	客户端 MAC	mac_0	UInt64		
	服务端 MAC	mac_1	UInt64		
	链路协议	eth_type	UInt16		显示为 0x+固定四个字符的 16 进制如 0x0806
	VLAN	vlan	UInt16		
网络层	客户端 IP	ip4_0/ip6_0	IPv4/IPv6		当 is_ipv4 为 0, 则取值为 ip6_x, 否则取值 ip4_x. 以下 IP 都同样处理.
	服务端 IP	ip4_1/ip6_1	IPv4/IPv6		
	IP 类	is_ipv4	UInt8		0: IPv4, 1: IPv6

传输层	网络协议	protocol	UInt8	
	隧道类型	tunnel_type	UInt8	是
	请求隧道ID	tunnel_tx_id	UInt32	是
	响应隧道ID	tunnel_rx_id	UInt32	是
	请求隧道源IP	tunnel_tx_ip4_0/tunnel_tx_ip6_0	IPv4/IPv6	是
	请求隧道目的IP	tunnel_tx_ip4_1/tunnel_tx_ip6_1	IPv4/IPv6	是
	响应隧道源IP	tunnel_rx_ip4_0/tunnel_rx_ip6_0	IPv4/IPv6	是
	响应隧道目的IP	tunnel_rx_ip4_1/tunnel_rx_ip6_1	IPv4/IPv6	是
	隧道IP类型	tunnel_is_ipv4	UInt8	
	客户端端口	client_port	UInt16	是
	服务端端口	server_port	UInt16	是
	客户端TCP标志位列表	tcp_flags_bit_0	UInt16	是

应用层 广域网 知识图谱	服务端TCP标志位列表	tcp_flags_bit_1	UInt16	是
	应用协议	l7_protocol	UInt8	0: 其他, 1: HTTP, 2: DNS
	客户端省份	province_0	String	
	服务端省份	province_1	String	
	客户端区域	region_id_0	UInt16	
	服务端区域	region_id_1	UInt16	
	客户端可用区	az_id_0	UInt16	
	服务端可用区	az_id_1	UInt16	
	客户端宿主机	host_id_0	UInt16	
	服务端宿主机	host_id_1	UInt16	
	客户端资源类型	l3_device_type_0	UInt8	

服务端资源类型	l3_device_type_1	UInt8
客户端资源	l3_device_id_0	UInt32
服务端资源	l3_device_id_1	UInt32
客户端容器节点	pod_node_id_0	UInt32
服务端容器节点	pod_node_id_1	UInt32
客户端容器命名空间	pod_ns_id_0	UInt16
服务端容器命名空间	pod_ns_id_1	UInt16
客户端容器工作负载	pod_group_id_0	UInt32
服务端容器工作负载	pod_group_id_1	UInt32
客户	pod_id_0	UInt32

端容 器 POD		
服务 端容 器 POD	pod_id_1	UInt32
客户 端容 器集 群	pod_cluster_id_0	UInt16
服务 端容 器集 群	pod_cluster_id_1	UInt16
客户 端 VPC	l3_epc_id_0	Int32
服务 端 VPC	l3_epc_id_1	Int32
客户 端网 口	epc_id_0	Int32
服务 端网 口	epc_id_1	Int32
客户 端子 网	subnet_id_0	UInt16
服务 端子 网	subnet_id_1	UInt16
客户 端资 源组 ID	group_ids_0	Array(UInt16)

列表				
服务端资源组 ID 列表	group_ids_1	Array(UInt16)		
客户端业务 ID 列表	business_ids_0	Array(UInt16)		
服务端业务 ID 列表	business_ids_1	Array(UInt16)		
流信息	流结束类型	close_type	UInt16	0: unknown, 所有不包含在取值范围的异常情况 1. tcp-fin: 正常结束 2. tcp-server-rst: 服务端其他异常 3. timeout: 超时 5. forced-report: 强制上报 7. client-syn-repeat: 客户端重复 SYN 8. server-half-close: 服务端半关 9. tcp-client-rst: 客户端其他异常 10. server-syn-ack-repeat: 服务端重复 SYN 11. client-half-close: 客户端半关

			13. client-source-port-reuse: 客户端端口复用
			15. server-reset: 服务端直接重置
			17. server-queue-lack: 服务端队列不足
			18. client-establish-other-rst: 客户端建连其他重置
			19. server-establish-other-rst: 服务端建连其他重置
流数 据来 源	flow_source	UInt16	0 原始流量, 1 sFlow, 2 NetFlow/NetStream v5
流日 志 ID	flow_id	UInt64	
采集 点	tap_type	UInt16	3 表示虚拟网络, 其他表示物理网络镜像
采集 网口 标识	tap_port	UInt32	显示为 0x+固定八个字符的 16 进制如 0x01234567
采集 器	vtap_id	UInt16	
客户 端二 层边 界	l2_end_0	UInt8	
服务 端二 层边 界	l2_end_1	UInt8	
客户	l3_end_0	UInt8	

指标量	服务端三层边界	l3_end_1	UInt8	
	开始时间	start_time	DateTime64	流的首包时间（微秒），或该流上条日志的 end_time
	结束时间	end_time	DateTime64	
	流持续时间	duration	UInt64	尾包和首包的时差（微秒）
	客户端发送包	packet_tx	UInt64	
	客户端接收包	packet_rx	UInt64	
	客户端发送字节	byte_tx	UInt64	
	客户端接收字节	byte_rx	UInt64	
	客户端发送网络层载荷	l3_byte_tx	UInt64	
	客户端接收网络层载荷	l3_byte_rx	UInt64	

客户端累计发送包	total_packet_tx	UInt64	
客户端累计接收包	total_packet_rx	UInt64	
客户端累计发送字节	total_byte_tx	UInt64	
客户端累计接收字节	total_byte_rx	UInt64	
L7 请求数	l7_request	UInt32	
L7 响应数	l7_response	UInt32	
累计TCP建连客户端时延	rtt_client_sum	Float64	是 微秒
TCP建连客户端时延次数	rtt_client_count	UInt64	
累计TCP建连服务	rtt_server_sum	Float64	是 微秒

端时延				
TCP 建连服务端时延次数	rtt_server_count	UInt64		
TCP 建连时延	rtt	Float64	是	微秒
累计 TCP 系统时延	srt_sum	Float64	是	微秒
TCP 系统时延次数	srt_count	UInt64		
累计传输层应用时延	art_sum	Float64	是	微秒
传输层应用时延次数	art_count	UInt64		
累计应用层应用时延	rrt_sum	Float64	是	微秒
应用层应用时延次数	rrt_count	UInt64		

最大 TCP 建连 客户 端时 延	rtt_client_max	UInt32	是 微秒
最大 TCP 建连 服务 端时 延	rtt_server_max	UInt32	是 微秒
最大 TCP 系统 时延	srt_max	UInt32	是 微秒
最大 传输 层应 用时 延	art_max	UInt32	是 微秒
最大 应用 层应 用时 延	rrt_max	UInt32	是 微秒
客户 端 TCP 重传 包	retans_tx	UInt32	
服务 端 TCP 重传 包	retrans_rx	UInt32	
客户 端 TCP 零窗	zero_win_tx	UInt32	

包					
服务端	zero_win_rx		UInt32		
TCP					
零窗包					
L7 客户端异常请求数	l7_client_error		UInt32		
L7 服务端异常响应数	l7_server_error		UInt32		
L7 服务端超时数	l7_server_timeout		UInt32		

HTTP 日志

区块	页面显示	存储字段名	存储类型	零值无意义	说明
网络层	客户端 IP	ip4_0/ip6_0	IPv4/IPv6		当 is_ipv4 为 0, 则取值为 ip6_x, 否则取值 ip4_x
	服务端 IP	ip4_1/ip6_1	IPv4/IPv6		
	IP 类型	is_ipv4	UInt8		0: IPv6, 1: IPv4
传输层	客户端口	client_port	UInt16	是	
	服务端口	server_port	UInt16	是	
应用	报文类型	type	UInt8		0: 请求, 1: 应答, 2: 会话

层

协议版本	version	UInt8		表示 HTTP 版本. 10: 1.0, 11: 1.1, 20: 2
请求方法	method	String	是	取值 GET、POST 等
代理客户端 IP	client_ip	String	是	X-Forwarded-For
主机名	host	String	是	
路径	path	String	是	
请求 ID	stream_id	String	是	仅 HTTP v2
TraceID	trace_id	String	是	解析 x-b3-traceid (zipkin), uber-trace-id (jeager)
响应码	status_code	String	是	

知识图谱

客户端区域	region_id_0	UInt16	
服务端区域	region_id_1	UInt16	
客户端可用区	az_id_0	UInt16	
服务端可用区	az_id_1	UInt16	
客户端宿主机	host_id_0	UInt16	
服务端宿主机	host_id_1	UInt16	
客户端资源类型	l3_device_type_0	UInt8	
服务端资源类型	l3_device_type_1	UInt8	
客户端资源	l3_device_id_0	UInt32	
服务端资源	l3_device_id_1	UInt32	

客户端容器节点	pod_node_id_0	UInt32
服务端容器节点	pod_node_id_1	UInt32
客户端容器命名空间	pod_ns_id_0	UInt16
服务端容器命名空间	pod_ns_id_1	UInt16
客户端容器工作负载	pod_group_id_0	UInt32
服务端容器工作负载	pod_group_id_1	UInt32
客户端容器 POD	pod_id_0	UInt32
服务端容器 POD	pod_id_1	UInt32
客户端容器集群	pod_cluster_id_0	UInt16
服务端容器集群	pod_cluster_id_1	UInt16
客户端 VPC	l3_epc_id_0	Int32
服务端 VPC	l3_epc_id_1	Int32
客户端网口 VPC	epc_id_0	Int32
服务端网口 VPC	epc_id_1	Int32
客户端子网	subnet_id_0	UInt16
服务端子网	subnet_id_1	UInt16
客户端资源组 ID 列	group_ids_0	Array(UInt16)

	表			
	服务端资源组 ID 列表	group_ids_1	Array(UInt16)	
	客户端业务 ID 列表	business_ids_0	Array(UInt16)	
	服务端业务 ID 列表	business_ids_1	Array(UInt16)	
流信息	流结束类型	close_type	UInt16	0: unknown, 所有不包含在取值范围的异常情况 1. tcp-fin: 正常结束
流信息	流日志 ID	flow_id	UInt64	
	采集点	tap_type	UInt16	3 表示虚拟网络, 其他表示物理网络镜像
	采集器	vtap_id	UInt16	
	开始时间	start_time	DateTime64	请求的首包时间戳 (微秒)
	结束时间	end_time	DateTime64	响应的首包时间戳 (微秒)
指标量	内容长度	content_length	Int64	Content-Length, 零存但没解析到不存
	响应延时	duration	UInt64	响应和请求的首包时差 (微秒)

DNS 日志

与 HTTP 类似，区别为： - 应用层不同，完整 DNS 应用层信息见下表 - 指标量中没有内容长度字段

区块	页面显示字段名	存储字段名	存储类型	零值无意义	说明
应用层	报文类型	type	UInt8		0: 请求, 1: 应答, 2: 会话
	请求 ID	id	UInt16		

查询域名	domain_name	String	是
查询类型	query_type	UInt16	是
响应码	answer_code	UInt16	当 type=0 时
解析地址	answer_addr	String	是

流量分发策略

分发策略相关 API

目录

分发策略

操作	HTTP Method
获取全部分发策略	GET
获取单个分发策略	GET
创建分发策略	POST
更新分发策略	PATCH
删除分发策略	DELETE

分发点

操作	HTTP Method
获取全部分发点	GET
获取单个分发点	GET
创建分发点	POST
更新分发点	PATCH
删除分发点	DELETE

API 详情

分发策略 API

获取全部分发策略

- URL:
/v1/deepflow/npb-policies/
- Response:
{
 "OPT_STATUS": "SUCCESS", // API 触发的操作是否成功，SUCCESS 表示成功，

其他值表示未成功

"WAIT_CALLBACK": false, // 是否为同步 API, 文档内所有 API 都是同步的

"TASK": null, // 该字段在文档中所有 API 中, 暂无意义

"DESCRIPTION": "", // API 触发的操作失败原因

"TYPE": "dict", // 暂无意义

"DATA": [{

"ID": 1,

"LCUUUID": "03b1b72b-3fe6-4d23-a1cd-4fe8ed966199",

"NAME": "yc-1000-100", // 名称

"STATE": 1, // 状态 0 禁用, 1 运行

"BUSINESS_ID": 1, // 暂时无用

"ACL_ID": 73214, // 供底层使用

"POLICY_ACL_GROUP_ID": 141, // 供底层使用

"PROTOCOL": null, // 协议

"SRC_PORTS": "22", // 采集点过滤规则端口

"DST_PORTS": "80", // 对端过滤规则端口

"SRC_SUB_TYPE": 11, // 采集点过滤规则端口

"DST_SUB_TYPE": 11, // 对端过滤资源子类型

"SRC_TYPE": 1, // 采集点过滤规则端口

"DST_TYPE": 1, // 对端过滤资源类型

"VNI": 100, // 流量标签

"NPB_TUNNEL_ID": 1, // 分发点 ID

"TAP_TYPE": 1, // 采集点

"PAYLOAD_SLICE": 1000, // 截断长度

"CREATED_AT": "2019-05-17 18:36:32", // 创建时间

"UPDATED_AT": "2019-05-18 10:53:29", // 修改时间

"SRC_IP": "192.168.1.1-192.168.2.3", // 采集点过滤规则 IP

"DST_IP": "10.30.100.39", // 对端过滤规则 IP

"SRC_EPC_ID": 2, // 采集点过滤规则 VPC ID

"DST_EPC_ID": 3, // 对端过滤规则 VPC ID

"SRC_VM_ID": null, // 采集点过滤规则云服务器 ID

"DST_VM_ID": null, // 对端过滤规则云服务器 ID

"SRC_GROUP_ID": null, // 采集点过滤规则资源组 ID

"DST_GROUP_ID": null, // 对端过滤规则资源组 ID

"SRC_POD_GROUP_ID": null, // 采集点过滤规则 POD 组 ID

"DST_POD_GROUP_ID": null, // 对端过滤规则 POD 组 ID

"SRC_POD_SERVICE_ID": null, // 采集点过滤规则容器服务 ID

"DST_POD_SERVICE_ID": null, // 对端过滤规则容器服务 ID

"SRC_POD_CLUSTER_ID": null, // 采集点过滤规则 POD 组所在集群 ID

"DST_POD_CLUSTER_ID": null // 对端过滤规则 POD 组所在集群 ID

}]

}

获取单个分发策略

- URL:

/v1/deepflow/npb-policies/{lcuuid}/

- Response:

```
{
  "OPT_STATUS": "SUCCESS", // API 触发的操作是否成功, SUCCESS 表示成功,
  其他值表示未成功
  "WAIT_CALLBACK": false, // 是否为同步 API, 文档内所有 API 都是同步
  的
  "TASK": null, // 该字段在文档中所有 API 中, 暂无意义
  "DESCRIPTION": "", // API 触发的操作失败原因
  "TYPE": "dict", // 暂无意义
  "DATA": {
    "ID": 1,
    "LCUUID": "03b1b72b-3fe6-4d23-a1cd-4fe8ed966199",
    "NAME": "yc-1000-100", // 名称
    "STATE": 1, //状态 0 禁用, 1 运行
    "BUSINESS_ID": 1, //暂时无用
    "ACL_ID": 73214, //供底层使用
    "POLICY_ACL_GROUP_ID": 141, //供底层使用
    "PROTOCOL": null, //协议
    "SRC_PORTS": "22", //采集点过滤规则端口
    "DST_PORTS": "80", //对端过滤规则端口
    "SRC_SUB_TYPE": 11, //采集点过滤规则端口
    "DST_SUB_TYPE": 11, //对端过滤资源子类型
    "SRC_TYPE": 1, //采集点过滤规则端口
    "DST_TYPE": 1, //对端过滤资源类型
    "VNI": 100, //流量标签
    "NPB_TUNNEL_ID": 1, //分发点 ID
    "TAP_TYPE": 1, //采集点
    "PAYLOAD_SLICE": 1000, //截断长度
    "CREATED_AT": "2019-05-17 18:36:32", //创建时间
    "UPDATED_AT": "2019-05-18 10:53:29", //修改时间
    "SRC_IP": "192.168.1.1-192.168.2.3", //采集点过滤规则 IP
    "DST_IP": "10.30.100.39", //对端过滤规则 IP
    "SRC_EPC_ID": 2, //采集点过滤规则 VPC ID
    "DST_EPC_ID": 3, //对端过滤规则 VPC ID
    "SRC_VM_ID": null, //采集点过滤规则云服务器 ID
    "DST_VM_ID": null, //对端过滤规则云服务器 ID
    "SRC_GROUP_ID": null, //采集点过滤规则资源组 ID
    "DST_GROUP_ID": null, //对端过滤规则资源组 ID
    "SRC_POD_GROUP_ID": null, //采集点过滤规则 POD 组 ID
    "DST_POD_GROUP_ID": null, //对端过滤规则 POD 组 ID
    "SRC_POD_SERVICE_ID": null, //采集点过滤规则容器服务 ID
    "DST_POD_SERVICE_ID": null, //对端过滤规则容器服务 ID
    "SRC_POD_CLUSTER_ID": null, //采集点过滤规则 POD 组所在集群 ID
  }
}
```

```

        "DST_POD_CLUSTER_ID": null    //对端过滤规则 POD 组所在集群 ID
    }
}

```

创建分发策略

- URL:

```
/v1/deepflow/npb-policies/{lcuuid}/
```

- 请求参数

字段	类型	描述	限制
BUSINESS_ID	int	业务 id	必填，仅支持：1
NAME	string	资源组名称	必填
STATE	int	状态	必填，0：禁用，1：启用
PROTOCOL	int	协议	非必填，范围：0~255
VNI	int	流量标签	非必填，范围：0~1023，默认：0
NPB_TUNNEL_ID	int	分发点 ID	必填
TAP_TYPE	int	采集点	必填
PAYLOAD_SLICE	int	payload 截断	非必填，范围：0~65535
SRC_PORTS	list of string	采集点过滤端端口	仅在协议指定为 TCP、UDP 时可填；支持单个端口及端口范围组合，举例：10,20-22、80；端口范围：1~65535
DST_PORTS	list of string	对端端口	仅在协议指定为 TCP、UDP 时可填；支持单个端口及端口范围组合，举例：10,20-22、80；端口范围：1~65535
SRC_TYPE	int	采集点过滤端类型	必填，1：虚拟化，2：容器，3：资源组
SRC_SUB_TYPE	int	采集点过滤端子类型	采集点过滤端类型为 1、2 时必填，为 3 时不填，11：IP，12：IP 段，13：子网，14：云服务器，21：POD 组
SRC_EPC_ID	int	采集点过滤端 VPC ID	采集点过滤端子类型为 11、12、13 时选填，为 14 时必填，其他不填
SRC_IP	string	采集点过滤端 IP/IP 段/	采集点过滤端子类型为 11、12、13 时必填，其他不填

		子网内容	
SRC_VM_ID	int	采集点过滤端云服务器 ID	采集点过滤端子类型为 14 时必填，其他不填
SRC_GROUP_ID	int	采集点过滤端资源组 ID	采集点过滤端类型为 3 时必填，其他不填
SRC_POD_CLUSTER_ID	int	采集点过滤端集群 ID	采集点过滤端类型为 2 时必填，其他不填
SRC_POD_GROUP_ID	int	采集点过滤端 POD 组 ID	采集点过滤端子类型为 21 时必填，其他不填
DST_TYPE	int	对端类型	1：虚拟化，2：容器，3：资源组
DST_SUB_TYPE	int	对端子类型	对端类型为 1、2 时必填，为 3 时不填，11：IP，12：IP 段，13：子网，14：云服务器，21：POD 组，22：服务
DST_EPC_ID	int	对端 VPC ID	对端子类型为 11、12、13 时选填，为 14 时必填，其他不填
DST_IP	string	对端 IP/IP 段 /子网内容	对端子类型为 11、12、13 时必填，其他不填
DST_VM_ID	int	对端云服务器 ID	对端子类型为 14 时必填，其他不填
DST_GROUP_ID	int	对端资源组 ID	对端类型为 3 时必填，其他不填
DST_POD_CLUSTER_ID	int	对端集群 ID	对端类型为 2 时必填，其他不填
DST_POD_GROUP_ID	int	对端 POD 组 ID	对端子类型为 21 时必填，其他不填
DST_POD_SERVICE_ID	int	对端服务 ID	对端子类型为 22 时必填，其他不填

- Request:

```
{
  "NAME": "npb_policy1",
  "STATE": 1,
  "BUSINESS_ID": 1,
  "PROTOCOL": 6,
  "VNI": 1,
  "NPB_TUNNEL_ID": 1,
  "TAP_TYPE": 1,
  "PAYLOAD_SLICE": 5,
```

```

"SRC_PORTS": "22-22",
"DST_PORTS": "22-22",
"SRC_TYPE": 1,
"SRC_SUB_TYPE": 11,
"SRC_EPC_ID": null,
"SRC_IP": '10.20.1.0',
"SRC_VM_ID": null,
"SRC_GROUP_ID": null,
"SRC_POD_CLUSTER_ID": null,
"SRC_POD_GROUP_ID": null,
"DST_TYPE": 1,
"DST_SUB_TYPE": 11,
"DST_EPC_ID": null,
"DST_IP": '10.20.1.0',
"DST_VM_ID": null,
"DST_GROUP_ID": null,
"DST_POD_CLUSTER_ID": null,
"DST_POD_GROUP_ID": null,
"DST_POD_SERVICE_ID": null
}

```

- Response:

```

{
  "DATA": {
    "NAME": "npb_policy1",
    "STATE": 1,
    "BUSINESS_ID": 1,
    "PROTOCOL": 6,
    "VNI": 1,
    "NPB_TUNNEL_ID": 1,
    "ACL_ID": 73214,
    "POLICY_ACL_GROUP_ID": 141,
    "TAP_TYPE": 1,
    "PAYLOAD_SLICE": 5,
    "SRC_PORTS": "22-22",
    "DST_PORTS": "22-22",
    "SRC_TYPE": 1,
    "SRC_SUB_TYPE": 11,
    "SRC_EPC_ID": null,
    "SRC_IP": '10.20.1.0',
    "SRC_VM_ID": null,
    "SRC_GROUP_ID": null,
    "SRC_POD_CLUSTER_ID": null,
    "SRC_POD_GROUP_ID": null,
    "DST_TYPE": 1,
    "DST_SUB_TYPE": 11,
    "DST_EPC_ID": null,
    "DST_IP": '10.20.1.0',
    "DST_VM_ID": null,

```

```

        "DST_GROUP_ID": null,
        "DST_POD_CLUSTER_ID": null,
        "DST_POD_GROUP_ID": null,
        "DST_POD_SERVICE_ID": null,
        "CREATED_AT": "2017-06-13 17:01:57",
        "UPDATED_AT": "2017-06-13 17:01:57",
        "LCUUUID": "a074a0a1-0525-4cb3-982a-f46c1b980832",
        "ID": 1
    },
    ...
}

```

更新分发策略

- URL:
/v1/deepflow/npb-policies/{lcuuid}/
- 请求参数（仅下发需要修改的字段）

字段	类型	描述	限制
TAP_TYPE	int	采集点	
STATE	int	状态	0：禁用，1：启用
PROTOCOL	int	协议	范围：0~255
PAYLOAD_SLICE	int	payload 截断	范围：0~65535
SRC_PORTS	list of string	采集点过滤端端口	仅在协议指定为 TCP、UDP 时可填；支持单个端口及端口范围组合，举例：10,20-22、80；端口范围：1~65535
DST_PORTS	list of string	对端端口	仅在协议指定为 TCP、UDP 时可填；支持单个端口及端口范围组合，举例：10,20-22、80；端口范围：1~65535
SRC_TYPE	int	采集点过滤端类型	1：虚拟化，2：容器，3：资源组
SRC_SUB_TYPE	int	采集点过滤端子类型	采集点过滤端类型为 1、2 时必填，为 3 时不填，11：IP，12：IP 段，13：子网，14：云服务器，21：POD 组
SRC_EPC_ID	int	采集点过滤端 VPC ID	采集点过滤端子类型为 11、12、13 时选填，为 14 时必填，其他不填
SRC_IP	string	采集点过滤	采集点过滤端子类型为 11、12、

		端 IP/IP 段/ 子网内容	13 时必填，其他不填
SRC_VM_ID	int	采集点过滤 端云服务器 ID	采集点过滤端子类型为 14 时必 填，其他不填
SRC_GROUP_ID	int	采集点过滤 端资源组 ID	采集点过滤端类型为 3 时必填， 其他不填
SRC_POD_CLUSTER_ID	int	采集点过滤 端集群 ID	采集点过滤端类型为 2 时必填， 其他不填
SRC_POD_GROUP_ID	int	采集点过滤 端 POD 组 ID	采集点过滤端子类型为 21 时必 填，其他不填
DST_TYPE	int	对端类型	1：虚拟化，2：容器，3：资源 组
DST_SUB_TYPE	int	对端子类型	对端类型为 1、2 时必填，为 3 时不填，11：IP，12：IP 段， 13：子网，14：云服务器，21： POD 组，22：服务
DST_EPC_ID	int	对端 VPC ID	对端子类型为 11、12、13 时选 填，为 14 时必填，其他不填
DST_IP	string	对端 IP/IP 段 /子网内容	对端子类型为 11、12、13 时必 填，其他不填
DST_VM_ID	int	对端云服务 器 ID	对端子类型为 14 时必填，其他 不填
DST_GROUP_ID	int	对端资源组 ID	对端类型为 3 时必填，其他不填
DST_POD_CLUSTER_ID	int	对端集群 ID	对端类型为 2 时必填，其他不填
DST_POD_GROUP_ID	int	对端 POD 组 ID	对端子类型为 21 时必填，其他 不填
DST_POD_SERVICE_ID	int	对端服务 ID	对端子类型为 22 时必填，其他 不填

- Request:

```
{
  "STATE": 1,
  "PROTOCOL": 6,
  "PAYLOAD_SLICE": 5,
  "SRC_PORTS": "22-22",
  "DST_PORTS": "22-22",
  "SRC_TYPE": 1,
  "SRC_SUB_TYPE": 11,
```

```

"SRC_EPC_ID": null,
"SRC_IP": '10.20.1.0',
"SRC_VM_ID": null,
"SRC_GROUP_ID": null,
"SRC_POD_CLUSTER_ID": null,
"SRC_POD_GROUP_ID": null,
"DST_TYPE": 1,
"DST_SUB_TYPE": 11,
"DST_EPC_ID": null,
"DST_IP": '10.20.1.0',
"DST_VM_ID": null,
"DST_GROUP_ID": null,
"DST_POD_CLUSTER_ID": null,
"DST_POD_GROUP_ID": null,
"DST_POD_SERVICE_ID": null
}

```

- Response:

```

{
  "DATA": {
    "NAME": "npb_policy1",
    "STATE": 1,
    "BUSINESS_ID": 1,
    "PROTOCOL": 6,
    "VNI": 1,
    "NPB_TUNNEL_ID": 1,
    "TAP_TYPE": 1,
    "ACL_ID": 73214,
    "POLICY_ACL_GROUP_ID": 141,
    "PAYLOAD_SLICE": 5,
    "SRC_PORTS": "22-22",
    "DST_PORTS": "22-22",
    "SRC_TYPE": 1,
    "SRC_SUB_TYPE": 11,
    "SRC_EPC_ID": null,
    "SRC_IP": '10.20.1.0',
    "SRC_VM_ID": null,
    "SRC_GROUP_ID": null,
    "SRC_POD_CLUSTER_ID": null,
    "SRC_POD_GROUP_ID": null,
    "DST_TYPE": 1,
    "DST_SUB_TYPE": 11,
    "DST_EPC_ID": null,
    "DST_IP": '10.20.1.0',
    "DST_VM_ID": null,
    "DST_GROUP_ID": null,
    "DST_POD_CLUSTER_ID": null,
    "DST_POD_GROUP_ID": null,
    "DST_POD_SERVICE_ID": null,
  }
}

```

```

        "CREATED_AT": "2017-06-13 17:01:57",
        "UPDATED_AT": "2017-06-13 17:01:57",
        "LCUUUID": "a074a0a1-0525-4cb3-982a-f46c1b980832",
        "ID": 1
    },
    ...
}

```

删除分发策略

- URL:

/v1/deepflow/npb-policies/{lcuuid}/

- Response:

```

{
  "DATA": {
    "LCUUUID": "61376cc5-4488-4795-9834-b680401343a5"
  },
  ...
}

```

分发点 API

获取全部分发点

- URL:

/v1/deepflow/npb-tunnels/

- Response

```

{
  "OPT_STATUS": "SUCCESS", // API 触发的操作是否成功，SUCCESS 表示成功，
  其他值表示未成功
  "TASK": null, // 该字段在文档中所有 API 中，暂无意义
  "DESCRIPTION": "", // API 触发的操作失败原因
  "TYPE": "NPB_TUNNEL", // 暂无意义
  "WAIT_CALLBACK": false, // 是否为同步 API，文档内所有 API 都是同步
  的
  "DATA": [{
    "ID": 1,
    "LCUUUID": "814e95d4-7b08-493b-9a1a-77db141995b0",
    "IP": "10.33.47.58", // 隧道端点 IP
    "TYPE": 1, // 隧道类型
    "POLICY_COUNT": 10, // 使用该分发点创建的分发策略个数
    "CREATED_AT": "2019-05-17 18:36:32", // 创建时间
    "UPDATED_AT": "2019-05-18 10:53:29", // 修改时间
    "NAME": "tunnel1" // 名称
  }
]

```

```
    }  
  }  
}
```

获取单个分发点

- URL:

/v1/deepflow/npb-tunnels//
- Response

{
 "OPT_STATUS": "SUCCESS", // API 触发的操作是否成功，SUCCESS 表示成功，其他值表示未成功
 "TASK": null, // 该字段在文档中所有 API 中，暂无意义
 "DESCRIPTION": "", // API 触发的操作失败原因
 "TYPE": "NPB_TUNNEL", // 暂无意义
 "WAIT_CALLBACK": false, // 是否为同步 API，文档内所有 API 都是同步的
 "DATA": {
 "ID": 1,
 "LCUUUID": "814e95d4-7b08-493b-9a1a-77db141995b0",
 "IP": "10.33.47.58", // 隧道端点 IP
 "TYPE": 1, //隧道类型
 "POLICY_COUNT": 10, //使用该分发点创建的分发策略个数
 "CREATED_AT": "2019-05-17 18:36:32", //创建时间
 "UPDATED_AT": "2019-05-18 10:53:29", //修改时间
 "NAME": "tunnel1" //名称
 }
}

创建分发点

- URL:

/v1/deepflow/npb-tunnels/{lcuuid}/
- 请求参数

字段	类型	描述
IP	string	隧道端点 IP 必填
NAME	string	名称 必填
TYPE	int	隧道类型 必填，0: VXLAN，1: ERSPAN

- Request:

{
 "NAME": "npb_tunnel1",
 "IP": "10.33.49.23",

```
    "TYPE": 1
  }
```

- Response:

```
{
  "DATA": {
    "ID": 1,
    "LCUUUID": "814e95d4-7b08-493b-9a1a-77db141995b0",
    "IP": "10.33.47.23",
    "TYPE": 1,
    "POLICY_COUNT": 0,
    "CREATED_AT": "2019-05-17 18:36:32",
    "UPDATED_AT": "2019-05-18 10:53:29",
    "NAME": "npb_tunnel1"
  },
  ...
}
```

更新分发点

- URL:

/v1/deepflow/npb-tunnels/{lcuuuid}/

- 请求参数（仅下发需要修改的字段）

字段	类型	描述	限制
IP	string	隧道端点 IP	
NAME	string	名称	
TYPE	int	隧道类型	0: VXLAN, 1: ERSPAN

- Request:

```
{
  "NAME": "npb_tunnel1",
  "IP": "10.33.49.23",
  "TYPE": 0
}
```

- Response:

```
{
  "DATA": {
    "ID": 1,
    "LCUUUID": "814e95d4-7b08-493b-9a1a-77db141995b0",
    "IP": "10.33.47.23",
    "TYPE": 0,
    "POLICY_COUNT": 0,
    "CREATED_AT": "2019-05-17 18:36:32",
```



```

        "UPDATED_AT": "2019-05-18 10:53:29",
        "NAME": "npb_tunnel1"
    },
    ...
}

```

删除分发点

- URL:

/v1/deepflow/npb-tunnels/{lcuuid}/

- Response:

```

{
  "DATA": {
    "LCUUID": "61376cc5-4488-4795-9834-b680401343a5"
  },
  ...
}

```

解析分发流量

封装格式

支持 VXLAN 或 ERSPAN 隧道封装，建议优先选择 ERSPAN 隧道封装。支持可选的内层 VLAN 封装，用于简化隧道对端汇聚设备或 TAP 交换机的配置。隧道封装格式和携带信息如下图所示：

原始包

原始ETH	原始IP	原始Payload

VXLAN封装

隧道ETH	隧道IP	隧道UDP	隧道VXLAN	原始ETH	可选VLAN	原始IP	原始Payload
			24 bit 序列号 24 bit 流量标签 1 bit 方向标签		12 bit 流量标签 1 bit 方向标签		

ERSPAN封装

隧道ETH	隧道IP	隧道GRE	隧道ERSPAN	原始ETH	可选VLAN	原始IP	原始Payload
		32 bit 序列号 32 bit 流量标签	48 bit 时间戳 10 bit 流量标签 1 bit 方向标签		12 bit 流量标签 1 bit 方向标签		

分发流量封装格式

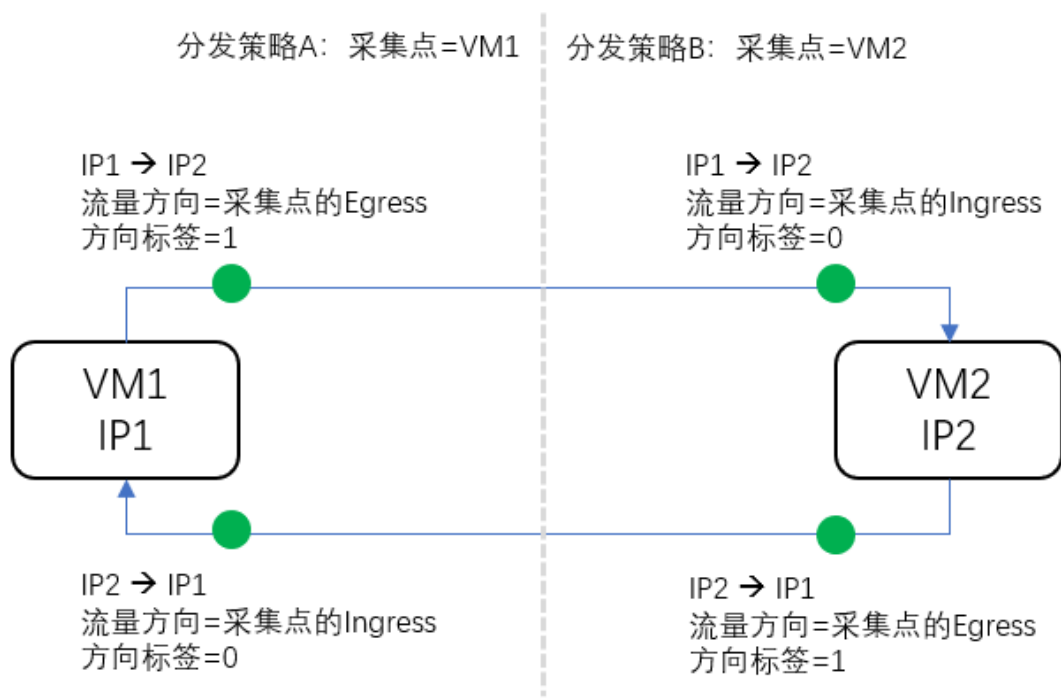
流量标签

支持流量标签和方向标签。流量标签通常用于区分不同 VPC 或不同业务的流量；方向标签通常用于分辨流量相对采集点的方向，也可用于区分客户端、服务端。

若分发策略中，采集点为服务端，那么流量接收方可根据方向标签来判断源 IP 或目的 IP 的类型：

- 方向标签=0：相对采集点的 Ingress 方向，即采集点为目的 IP，

因此目的 IP 为服务端 IP - 方向标签=1：相对采集点的 Egress 方向，即采集点为源 IP，因此源 IP 为服务端 IP



分发流量方向标签

流量解封装参考

内层 VLAN 封装

VLAN 头部格式参考 [Wikipedia](#)。

当接收端 TAP 交换机或分流设备仅有隧道解析能力，无法根据隧道头部设置正确的 VLAN 及 PCP 标签时，DeepFlow 采集器可设置为在原始包以太头后部插入一个 VLAN 标签。DeepFlow 采集器使用的标记字段包括： - 使用 VID 记录流量标签的低 12 比特 - 使用 PCP 记录流量采集的位置，也可理解为在采集点观测到的流量方向 - PCP 最高位永远为 1 - PCP 最低位与方向标签的含义一致

```
> Frame 1: 1058 bytes on wire (8464 bits), 1058 bytes captured (8464 bits)
> Ethernet II, Src: VMware_b3:70:ff (00:50:56:b3:70:ff), Dst: VMware_b3:11:8d (00:50:56:b3:11:8d)
> 802.1Q Virtual LAN, PRI: 5, DEI: 0, ID: 91
  101. .... = Priority: Voice, < 10ms latency and jitter (5)
  ...0 .... = DEI: Ineligible
  ... 0000 0101 1011 = ID: 91
  Type: IPv4 (0x0800)
> Internet Protocol Version 4, Src: 10.50.101.29, Dst: 10.50.101.31
> Transmission Control Protocol, Src Port: 47367, Dst Port: 0, Seq: 0, Len: 1000
> Data (1000 bytes)
```

VLAN 包格式

[VLAN 样包下载](#)（请右键另存文件，或在浏览器新标签页中打开链接）

是否携带内层 VLAN 头是可配置的，默认不携带内层 VLAN，在系统-采集器-配置中修改包分发配置参数-内层附加头为 802.1Q 可开启内层 VLAN 封装。

VXLAN 封装

VXLAN 隧道封装格式参考 [RFC 7348](#)。

DeepFlow 采集器使用的标记字段包括：- 设置 Flags 为 0xff 来区分于业务网络的 VXLAN 流量 - 使用 Flags 之后的 24 bit 预留位（下图的 Group Policy ID）记录序列号，辅助接收端判断隧道丢包 - 使用 24 bit VNI（VXLAN Network Identifier）字段记录流量标签 - 使用 VNI 之后的一个保留字节的最低比特记录方向标签

```
> Frame 1: 1108 bytes on wire (8864 bits), 1108 bytes captured (8864 bits)
> Ethernet II, Src: VMware_b3:8d:34 (00:50:56:b3:8d:34), Dst: VMware_b3:2a:b9 (00:50:56:b3:2a:b9)
> Internet Protocol Version 4, Src: 172.29.220.200, Dst: 172.29.220.210
> User Datagram Protocol, Src Port: 49152, Dst Port: 4789
< Virtual eXtensible Local Area Network
  > Flags: 0x0800, VXLAN Network ID (VNI)
    Group Policy ID: 8
    VXLAN Network Identifier (VNI): 90
    Reserved: 1
  > Ethernet II, Src: VMware_b3:70:ff (00:50:56:b3:70:ff), Dst: VMware_b3:11:8d (00:50:56:b3:11:8d)
< 802.1Q Virtual LAN, PRI: 5, DEI: 0, ID: 90
  101. .... = Priority: Voice, < 10ms latency and jitter (5)
  ...0 .... = DEI: Ineligible
  .... 0000 0101 1010 = ID: 90
  Type: IPv4 (0x0800)
> Internet Protocol Version 4, Src: 10.50.101.29, Dst: 10.50.101.31
> Transmission Control Protocol, Src Port: 1370, Dst Port: 0, Seq: 0, Len: 1000
> Data (1000 bytes)
```

VXLAN 包格式

[VXLAN 样包下载](#)（请右键另存文件，或在浏览器新标签页中打开链接）

ERSPAN 封装

GRE 头部格式参考 [RFC 2890](#)。ERSPAN 头部格式参考 [IETF ERSPAN Draft 03](#)。

DeepFlow 采集器使用的标记字段包括：- 使用 32 bit 的 GRE Key 字段记录流量标签 - 使用 32 bit 的 GRE Sequence Number 字段记录序列号字段，辅助接收端判断隧道丢包 - 使用 10 bit 的 ERSPAN SpanID 字段记录流量标签的低 10 比特，当接收端无法解析 GRE Key 时刻使用此字段 - 使用 48 bit 的 ERSPAN Timestamp + SGT（Security Group Tag）记录纳秒时间戳的高 48 比特 - 使用 1 bit 的 ERSPAN Direction 字段记录方向标签

```
> Frame 1: 1116 bytes on wire (8928 bits), 1116 bytes captured (8928 bits)
> Ethernet II, Src: VMware_b3:8d:34 (00:50:56:b3:8d:34), Dst: VMware_b3:2a:b9 (00:50:56:b3:2a:b9)
> Internet Protocol Version 4, Src: 172.29.220.200, Dst: 172.29.220.210
> Generic Routing Encapsulation (ERSPAN III)
  > Flags and Version: 0x3000
    Protocol Type: ERSPAN III (0x22eb)
    Key: 0x0000005b
    Sequence Number: 962995
  > Encapsulated Remote Switch Packet ANalysis Type III
    0010 .... = Version: Type III (2)
    ... 0000 0000 0000 = Vlan: 0
    000. .... = COS: 0
    ...0 0... = Bad/Short/Oversized: Good or unknown integrity (0)
    ....0... = Truncated: Not truncated (0)
    ....00 0101 1011 = SpanID: 91
    0001 0110 0001 1111 0001 0101 1010 0010 = Timestamp: 371135906
    0101 1000 0001 0110 = Security Group Tag: 22550
    0... = Has Ethernet PDU: 0
    .000 00.. = Frame Type: Ethernet (0)
    ....00 0000 = Hardware ID: 0
    ....1... = Direction: Egress
    ....11. = Timestamp granularity: Custom granularity (3)
    ....0 = Optional Sub headers: 0
  > Ethernet II, Src: VMware_b3:70:ff (00:50:56:b3:70:ff), Dst: VMware_b3:11:8d (00:50:56:b3:11:8d)
  > 802.1Q Virtual LAN, PRI: 5, DEI: 0, ID: 91
    101. .... = Priority: Voice, < 10ms latency and jitter (5)
    ...0 .... = DEI: Ineligible
    ...0000 0101 1011 = ID: 91
    Type: IPv4 (0x0800)
  > Internet Protocol Version 4, Src: 10.50.101.29, Dst: 10.50.101.31
  > Transmission Control Protocol, Src Port: 13192, Dst Port: 0, Seq: 0, Len: 1000
  > Data (1000 bytes)
```

ERSPAN 包格式

[ERSPAN 样包下载](#)（请右键另存文件，或在浏览器新标签页中打开链接）

最佳实践

利用内层 VLAN 封装简化 TAP 设备配置

当客户已有具备 VXLAN 或 ERSPAN 隧道解封能力的 TAP 设备时，采集器可发送流量至 TAP 设备，再由 TAP 设备将流量分发给其他分析工具。这种情况下推荐打开内层 VLAN 封装，使得分发流量中同时携带外层隧道封装和内层 VLAN 封装。此时 TAP 设备上可直接配置一条对所有隧道封装流量的解封策略，无需对流量做任何额外的标记操作即可直接转发给后端分析工具。分析工具通过 VLAN 头中的 ID 识别分发的云网流量。

利用 ERSPAN 封装提升性能分析工具精准度

若采集器和分析工具之间需要经过复杂的交换机、路由器甚至防火墙，此时路径中的多网元、网元上的多接口、接口内的多个转发队列都有可能会导致分发流量在传输过程中出现不一致的时延、抖动、甚至乱序。这种场景下推荐使用 ERSPAN 封装，DeepFlow 采集器将会在 ERSPAN 头部中记录流量在采集位置出现时的高精度时间

戳，分析工具可利用该时间戳精准还原报文时序，避免受到乱序干扰，同时也能由此计算更加精确的网络性能指标数据。

网络拓扑

API 文档

请联系云杉网络服务人员获取 API 文档

Grafana 插件

grafana_df_plugin 插件部署

- 注意，grafana-server 的版本需要高于 6.5.0
- 在客户的 grafana-server 机器上安装
 - 需要确认：
 - 可使用 rpm 命令
 - grafana 的插件目录为默认地址 /var/lib/grafana/plugins（可以在 grafana 默认配置文件/etc/grafana/grafana.ini 中搜索关键字 plugins=）
 - 当上述两条均满足时，直接从 deepflow 容器化部署 iso 包 extra-artifacts 目录中将 df-grafana-plugin, df-grafana-panel, df-grafana-querier 中的包拷贝至客户机器，使用 rpm 安装。
 - 当上述两条不满足时，需要联系研发根据具体场景直接以代码文件形式部署。
- 安装完毕后：
 - 初次安装时，进入 datasource 插件安装目录 /var/lib/grafana/plugins/deepflow，将 plugin.json 中 DEEPFLOW_DOMAIN 替换为 DeepFlow 主控制器的地址，注意协议（http/https）一致，并保证 grafana 可以访问该地址。注意，当 DeepFlow 使用 https+IP 形式，且无有效证书时，会无法正常访问，可通过开启 DeepFlow 的 http 端口来使用，参见控制器中 /etc/openresty/conf/conf.d/http.conf。
 - 非初次安装时，不会替换原 plugin.json 文件，新配置文件以 plugin.json.new 形式存在。panel 插件不需要改动 plugin.json。
 - 重启 grafana-server: `systemctl restart grafana-server`
- 高版本 grafana（目前发现>=8.0.0）中，配置-plugins 中提示插件 unsigned 不可用时：
 - 在/etc/grafana/grafana.ini 中最后添加如下配置（注意，如果已有 plugins 这一节，直接添加加下面一行即可。）：[plugins]
allow_loading_unsigned_plugins = yunshan-deepflow-datasource,yunshan-df-relation-panel,yunshan-metaflow-datasource

- 注意，这三个包可以独立安装，对应的配置中只需要写相应的名称。
- 重启 grafana-server: `systemctl restart grafana-server`

Grafana panel 插件

在 Grafana->Panel->Visualization 中可看到 DeepFlow ServiceGraph，该插件只能和 DeepFlow datasource 插件以及 DeepFlow 拓扑图 API 配合使用。- 注意：仅展现 DeepFlow 搜索条件中资源与资源外、广域网之间的访问路径，资源集合内部的访问关系不会展现

Data Sources 设置

- DeepFlow
 - 选择 Data Source 类型:
 - DeepFlow
 - 配置 Data Source
 - API Token: Token 的获取方式详情见 [TOKEN 获取-1/2 章节](#)
 - 注意，由于 grafana 自身的 bug #22552，在清除已设置的 API Token 后，再次刷新还会显示为已配置，此时重新设置 token 即可。注意需要保存。
- MetaFlow
 - 选择 Data Source 类型:
 - MetaFlow
 - 配置 Data Source
 - Request Url: 配置 queier 对应的后端服务请求地址
 - API Token: DeepFlow 内使用时, 若需要 token 验证, 在此处填写, Token 的获取方式详情见 [TOKEN 获取-1/2 章节](#)
 - 注意事项:
 - 在 DeepFlow 内使用时, 只能查询默认区域的数据
 - 交互相关:
 - 画折线图时, 需要选中 group by, 并设置 interval, 且在 select 中选中 metric(目前仅支持选中 1 项)
 - 若 select/ group by 选中 resource 类型的 tag, 则自动添加 resource_id
 - 若 where 中, 选中 resource 类型的 tag, 当是 =, !=, IN, NOT IN 时, 则自动转换为 resource_id
 - group by/ order by 会自动查找 select 中是否用过且有 as 的项, 有的话直接复用 as 后的名称, 若存在多个匹配项, 使用第一个有 as 的

- 使用 group by 时, select 内 tag 的可选项为, group by 中选中的 tag 的集合
- 使用 group by 时, metric 可配置 func, param
- 使用 group by 时, order by 内 metric 的可选项为, select 中选中的 metric 的集合
- 若未填写 limit, 则禁用 offset

Dashborad 配置

- 添加 Panel 的 Query 时，查询语句填写：
 - DeepFlow 全景图确认需要添加到 Dashborad 的子视图
 - 点击查看 API：点击查看后在末尾会显示搜索条件的文字描述，用于直观解释该 API 的作用。
 - 点击复制按钮
 - 粘贴到 Grafana->Dashborad->Panel->Query 中
 - 时间范围、时间粒度以 Grafana Panel 为依据
 - 如需 Grafana 的 Variable 控制 DeepFlow 子视图对应的 Panel，有如下限制：
 - DeepFlow 中子视图搜索条件引用的模板变量与 Grafana 的 Variable 同名
 - 目前支持如下类型的 DeepFlow 模板变量，对应关系如下：

DeepFlow 模板变量类型	Grafana Variable Type	Grafana Variable Values	备注
容器集群	无限制	\$容器集群名称	-
命名空间	无限制	\$命名空间名称	-
工作负载	无限制	\$工作负载名称	-
容器 POD	无限制	\$容器 POD 名称	-
资源分组	Custom	value : key	value : key 加入即可，见后续章节。例如：容器集群：pod_clust

网络协议 Custom value : key

应用协议 Custom value : key

服务端端口 Text box 界面直接输入

HTTP 日志-请求方法 Custom GET,POST,HEAD,PUT,DELETE,OPTIONS,TRACE,CONNECT,PATCH

HTTP 日志-主机名 Text box 界面直接输入

HTTP 日志-路径 Text box 界面直接输入

HTTP 日志-响应码 Text box 界面直接输入

er, 命名空间 :
pod_ns,
工作负载 :
pod_group
, 容器
POD : pod
value : key
加入即可, 见后续章节。
例如:
TCP : 6,
UDP : 17
value : key
加入即可, 见后续章节。
例如:
HTTP : 1,
DNS : 2

-

-

-

-

- DeepFlow 子视图与 Grafana Panel 的映射关系如下:

DeepFlow 子视图	Grafana Panel	备注
TOP N 折线图	Graph	
TOP N 折线图	Stat	

TOP N 折线图	Gauge
TOP N 折线图	Bar gauge
TOP N 折线图	Table
TOP N 折线图	Heatmap
折线图	Graph
折线图	Stat
折线图	Gauge
折线图	Bar gauge
折线图	Table
折线图	Heatmap
资源变更事件图	Graph
资源变更事件图	Stat
资源变更事件图	Gauge
资源变更事件图	Bar gauge
资源变更事件图	Table
资源变更事件图	Heatmap
流量拓扑	DeepFlow ServiceGraph

Grafana Variable Values 的映射关系

以下 value 名称都支持修改，表格只是推荐名称

资源分组

key	value
auto_service	自动分组-服务聚合
auto_pod_group	自动分组-工作负载聚合
auto	自动分组-全展开
host	宿主机
vm	云服务器
pod_cluster	容器集群
pod_ns	命名空间

pod_node	容器节点
pod_group	工作负载
pod	容器 POD
l3_epc	VPC
subnet	子网
ip	IP 地址
business	业务
group	资源组
region	区域
az	可用区

网络协议

key	value
6	TCP
17	UDP

如需其他网络协议，key 值为协议编号，value 值可参考文档的关键字，[参考文档](#) - 注意：协议编号=0 时，DeepFlow 语义与标准协议定义不一致，语义表达的为非 IP，因此 value 建议为非 IP

应用协议

key	value
1	HTTP
5	Dubbo
3	MySQL
4	Redis
6	Kafka
2	DNS
7	未知

API 权限获取

TOKEN 获取

建议： K8s 部署方式下，所有操作都是在 front-end 的 pod 下执行，获取 fron-end 的 pod 名称以及进入 pod 方式如下：

```
kubectl -n deepflow get pod -o wide |grep front-end
kubectl -n deepflow exec -it front-end-xxxx-xxxx-xxxx /bin/sh
```

获取永久授权码

- 系统内部访问（rpm 部署方式）：

```
curl http://127.0.0.1:20822/valid_token/<email>/<pwd>
```

- 系统内部访问（K8s 部署方式）：

```
curl http://fauths:20822/valid_token/<email>/<pwd>
```

返回值：

```
{
  "OPT_STATUS": "SUCCESS",
  "DATA": "gAAAAABexhfQWERDQWESDFShVjc1JhRDNxbsF2btXVND-i8Qbo_xAqe0IQ6N
KJaH13Yg9MmlwoDbYnOtI7Mzpnjdqgv1iR_gKQhNCdaFO1d3w2YkSg_ZaT8rMpXEN82oSnI
-MaHU4iUFRpgc9MA_s-fi2dfs9bMtYZw04MMkAsDF6LZv4u6nvc07pIrFGDiOKaBYd5A71r8
wHfryXyAQWERAfQDSDFQW5w_ZLEHGg==",
  "DESCRIPTION": "SUCCESS"
}
```

DATA 值为永久有效的 access_token 值

建议： 说明： 请求的 url 中的 <email> 和 <pwd> 请换成需要获取永久有效 token 的账户信息

获取短时授权码

- 系统内部访问（rpm 部署方式）

```
curl -XPOST -H"content-type: application/json" -d'{"account_type":"deep
flow","grant_type": "password","password": "dev-pwd","email": "dev@yuns
han.net.cn"}' http://127.0.0.1:20822/login
```

- 系统内部访问（K8s 部署方式）

```
curl -XPOST -H"content-type: application/json" -d'{"account_type":"deep
flow","grant_type": "password","password": "dev-pwd","email": "dev@yuns
han.net.cn"}' http://fauths:20822/login
```

- 系统外部访问

```
curl -XPOST -H"content-type: application/json" -d'{"account_type":"deep
flow","grant_type": "password","password": "dev-pwd","email": "dev@yuns
han.net.cn"}' https://domain/api/fauths/login
```

返回：

```
{
  "OPT_STATUS": "SUCCESS",
  "DATA": {
    "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOiJlNzE4MjMwZAsIm1",
    "refresh_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOiJlNzE4MjMwZAsIm1",
    "expires_in": 3600,
    "token_type": "password",
  }
}
```

```

"email": "dev@yunshan.net.cn",
"license": true
},
"DESCRIPTION": ""
}

```

access_token 为短期有效 token。expires_in 为默认过期时间，单位：秒。
refresh_token 为刷新 access_token 必备的 token，默认该 token 过期时间为 5 天。

刷新短时授权码

- 系统内部访问（rpm 部署方式）

```

curl -XPOST -H"content-type: application/json" -d'{"grant_type": "refresh_token", "refresh_token": "eyJ0eXAiOiJKV1QiLCJ"}' http://127.0.0.1:20822/refresh_token

```

- 系统内部访问（K8s 部署方式）

```

curl -XPOST -H"content-type: application/json" -d'{"grant_type": "refresh_token", "refresh_token": "eyJ0eXAiOiJKV1QiLCJ"}' http://fauths:20822/refresh_token

```

- 系统外部访问

```

curl -XPOST -H"content-type: application/json" -d'{"grant_type": "refresh_token", "refresh_token": "eyJ0eXAiOiJKV1QiLCJ"}' https://domain/api/fauths/refresh_token

```

返回：

```

{
"OPT_STATUS": "SUCCESS",
"DATA": {
"access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI",
"refresh_token": "eyJ0eXAiOiJKV1QiLCJhbGciOi",
"expires_in": 3600,
"token_type": "refresh_token"
},
"DESCRIPTION": ""
}

```

access_token 为短期有效 token。expires_in 为默认过期时间，单位：秒。
refresh_token 为刷新 access_token 必备的 token，默认该 token 过期时间为 5 天。

校验授权码获取用户信息

- 系统内部访问（rpm 部署方式）

```

curl -XPOST -H"content-type: application/json" -H"Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI" http://127.0.0.1:20822/auth

```

- 系统内部访问（k8s 部署方式）

```
curl -XPOST -H"content-type: application/json" -H"Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI" http://fauths:20822/auth
```

Authorization 由两部分组成，一部分是 Bearer 一部分是 登录时获取到的 access_token,拼接时中间有一个 空格连接

返回:

```
{
  "OPT_STATUS": "SUCCESS",
  "DATA": {
    "id": 1000,
    "useruuid": "8a821252-7f9f-4f38-ac94-a4876d5d13d2", "username": "dev",
    "user_type": 2,
    "real_user_type": 2,
    "state": 1,
    "email": "dev@yunshan.net.cn",
    "login_time": 1571820386,
    "company": "yunshan",
    "phone_num": ''
  },
  "DESCRIPTION": ""
}
```

注销授权码

- 系统内部访问 (rpm 部署方式)

```
curl -XPOST -H"content-type: application/json" -H"Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI" http://127.0.0.1:20822/logout
```

- 系统内部访问 (k8s 部署方式)

```
curl -XPOST -H"content-type: application/json" -H"Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI" http://fauths:20822/logout
```

- 外部访问

```
curl -XPOST -H"content-type: application/json" -H"Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI" https://domain/api/fauths/logout
```

Authorization 由两部分组成，一部分是 Bearer 一部分是 登录时获取到的 access_token,拼接时中间有一个 空格连接

返回:

```
{
  "OPT_STATUS": "SUCCESS",
  "DATA": null,
  "DESCRIPTION": ""
}
```

案例

部分接口需要在 header 中添加用户信息才可以访问，案例如下：

- 系统内部访问（rpm 部署方式）
`curl -XGET -H"content-type: application/json" -H"X-User-Id: 1" -H"X-User-Type:2" http://127.0.0.1:20840/v1/vms`
- 系统内部访问（K8S 部署方式）
`curl -XGET -H"content-type: application/json" -H"X-User-Id: 1" -H"X-User-Type:2" http://fauths:20840/v1/vms`
- 外部访问
`curl -XGET -H"content-type: application/json" -H"Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI" https://domain/api/talker/v1/vms`

X-User-Id 和 X-User-Type 来源是校验授权码接口中返回的用户信息

告警

告警事件

告警相关API

资源	操作	HTTP Method	Filters
告警事件	获取全部告警事件	GET	page_index, page_size, business_id, policy_id, app_type

告警事件

GET

- URL:
 - /v1/deepflow/events/
 - /v1/deepflow/start_time//end_time//
- filters:
 - page_index: 分页页数
 - page_size: 分页大小
 - filter_by: 名称正则表达式
 - business_id: 对象 id
 - app_type: app 类型
 - policy_id: 策略 id
 - sub_view_id: 子视图 id
- Response:

```

{
  "OPT_STATUS": "SUCCESS",
  "DESCRIPTION": "",
  "DATA": {
    "hits": [
      {
        "id": "226-1607581980",
        "status": "end",          # 状态: start: 已开始; end: 已结束
        "timestamp": 1607581980, # 事件开始时间
        "policy_id": 226,         # 告警策略 ID
        "policy_name": "test",    # 告警策略名称
        "policy_app_type": 2,     # 告警应用类型: 1, 系统; 2: 全
        "policy_email": "x@y.com", # 告警发送邮件地址
        "policy_level": 0,        # 告警级别: 0: 低, 1: 中, 2: 高
        "policy_data_level": "1m", # 告警使用的数据源
        "policy_target_line": "\u865a\u62df\u7f51\u7edc",
        "policy_target_full": "\u91c7\u96c6\u70b9=\u865a\u62df\u7f51\u7edc",
        "policy_target_field": "avg_bps_tx", # 告警监控的指标量
        "sub_view_id": 2686,             # 字数图 ID
        "sub_view_name": "xxxx",         # 字数图名称
        "trigger_condition": "\u53d1\u9001\u6bd4\u7279\u5747\u503c\u5927\u4e8e0.0\u6bd4\u7279\u79d2\u6216\u8005\u5c0f\u4e8e0.0\u6bd4\u7279\u79d2", #触发条件
        "trigger_additional": "",
        "end_time": 1607589420           # 事件结束时间
      }
    ],
    "PAGE": {
      "INDEX": 1,
      "TOTAL": 24,
      "SIZE": 1
    }
  }
}

```

景图

系统

采集器

采集器相关 API

目录

采集器

采集器

操作	HTTP Method
获取全部采集器	GET

API 详情

采集器 API

获取全部采集器

- URL:

/v1/vtaps/

- Response:

```
{
  "OPT_STATUS": "SUCCESS", // API 触发的操作是否成功, SUCCESS 表示成功,
  其他值表示未成功
  "WAIT_CALLBACK": false, // 是否为同步 API, 文档内所有 API 都是同步的
  "TASK": null, // 该字段在文档中所有 API 中, 暂无意义
  "DESCRIPTION": "", // API 触发的操作失败原因
  "TYPE": "VTAP", // 暂无意义
  "DATA": [
    {
      "ID": 5522,
      "LCUUUID": "99c8e75c-e40e-46ac-bc6b-67f0aa765006",
      "NAME": "analyzer23", // 名称
      "STATE": 1, // 状态 0 失联, 1 运行
      "ENABLE": 1, // 使能状态 0 禁用 1 启用
      "LAUNCH_SERVER": "10.33.49.23", // 运行环境 IP
      "TYPE": 6, // 类型 1: KVM 2: ESXI 3: Worklod-V 5: Workload-P
      6: 专属服务器 7: 容器-P 8: 容器-V 9: 隧道解封装
      "CTRL_IP": "10.33.49.23", // 控制 IP
      "ANALYZER_IP": "10.33.49.23", // 数据节点
      "CONTROLLER_IP": "10.33.49.23", // 控制器
      "CUR_ANALYZER_IP": "10.33.49.23", // 当前访问数据节点
      "CUR_CONTROLLER_IP": "10.33.49.23", // 当前访问控制器
      "SYNCED_CONTROLLER_AT": "2020-12-10 14:15:11", // 控制器同步时
      间
      "SYNCED_ANALYZER_AT": "2020-12-10 14:05:16", // 数据节点通信时
      间
      "BOOT_TIME": 1607508184, // 启动时间
      "REVISION": "1061", // 软件版本
      "EXCEPTIONS": [ // 异常信息
        256
      ],
      "VTAP_GROUP_LCUUUID": "e8f80e94-7885-11e9-976d-f48e38c2174c",
    }
  ]
}
```



```

        "VTAP_GROUP_NAME": "默认", // 资源组
        "AZ": "a435cb81-df07-537d-a772-5a234f5d3bff",
        "AZ_NAME": "openstack-32", // 可用区
        "REGION": "25aeb190-6242-4497-9a5a-e36c10fcba3f",
        "REGION_NAME": "北京办公室", // 区域
        "COMPLETE_REVISION": "bb143edf4ce43d8c58667d04e330dad756a4aaf
4", // 完整软件版本
        "CPU_NUM": 24, // 总 CPU(核)
        "MEMORY_SIZE": 67290140672, // 总内存
        "ARCH": "x86_64", // 体系架构
        "ARCH_TYPE": 1,
        "OS": "centos 7.5.1804", // 操作系统
        "OS_TYPE": 1,
        "KERNEL_VERSION": "4.19.17" // 内核版本
    },
}
]
}

```

故障处理

控制器告警

控制器负载高

- 告警触发条件：控制器负载 5 分钟持续达到限制的 70%
- 计算方法：CentOS 系统中通过 `top` 命令查询到的最近 1 分钟 load 值与服务器 CPU 逻辑核数的比值
- 处理方法：
 - 点击告警事件中的控制器主机名，或访问系统管理-控制器页面并点击控制器名字
 - 查看控制器最近 24 小时系统负载曲线
 - 若超过 5%的时间达到服务器 CPU 逻辑核数量的 70%，则需要服务器扩容
 - 否则，查看同时是否有其他告警事件，解决其他告警即可

控制器磁盘空间不足

- 告警触发条件：控制器任何一块磁盘使用空间大于 70%
- 处理方法：
 - 点击告警事件中的控制器主机名，或访问系统管理-控制器页面并点击控制器名字
 - 查看控制器最近 24 小时磁盘用量曲线，评估是否需要磁盘或服务器扩容

控制器失联

- 告警触发条件：主控制器无法通过 RESTful API 连通故障控制器
- 计算方法：
 - 主控制周期调用 `http://{host}:20014/v1/health/` 探测，若收到回复则认为正常
 - 若超过一分钟从未收到回复信息，则认为异常
 - 主控制器每隔 3 秒扫描一次所有控制器和数据节点
- 处理方法：
 - 运维人员收到告警后应当尽快通过带外的方式登录控制器进行恢复

注意： 控制器失联时，采集器自动切换连接的控制器，默认切换时间为 1 分钟（由系统管理-采集器-配置页面的最长同步间隔决定）。

注意： 若有控制器或数据节点失联，控制器探测的时间间隔可能由于等待超时大于 3 秒。

建议： 默认情况下通过 22 端口探测 SSH 服务，若部署环境使用其他 SSH 端口需要修改 `/etc/manager.yaml` 中的 `check_port` 并重启 `manager` 进程。

采集器告警

采集器失联

- 告警触发条件：采集器被控制器标记为失联
- 计算方法：当控制器连续 5 个同步周期没有接收到采集器的同步消息时被认为是失联，同步周期为系统-采集器-配置中设置的最长同步间隔
- 处理方法：
 - 在控制器上可进行的排查
 - 通过抓包确认采集器是否正常请求了控制器的 TCP 20035（当打开 TLS 时为 20135）端口
 - 若采集器在正常请求，打开系统-采集器-配置中的日志发送，并在控制器的 `/var/log/messages` 中确认采集器是否有 WARN 或 ERR 日志
 - 登录采集器所在服务器进行排查
 - `systemctl status trident` 确认进程是否在运行、及是否设置为开机自启动（enable）
 - 通过 `telnet` 确认是否可请求控制器的 TCP 20035（当打开 TLS 时为 20135）端口
 - 观察 `trident` 的 CPU 开销，确认是否由于 `cgroup` 或 `k8s` 配置导致 CPU 受限、进而导致请求控制器变慢

采集器丢包

- 告警触发条件：采集器进程内部队列丢包
- 处理方法：
 - trident_dispatcher_kernel_drops
 - trident_collect_sender_send_dropped
 - 在控制器执行 `mt trisolaris.check ctrl_ip=172.16.1.103 | grep analyzer_ip` 检查配置的数据节点 IP 是否正确
 - 在控制器执行 `trident-ctl rpc basicConfig --ip <vtap-ip> | grep analyzer_ip` 检查下发的数据节点 IP 是否正确
 - 在控制器执行 `trident-ctl rpc basicConfig --ip <vtap-ip> | grep socket_type` 确认通信端口的类型，0 为 UDP，1 为 TCP
 - 在对应的数据节点上检查端口是否正常监听
 - `droplet: netstat -anp | grep 20033`
 - 通信端口为 TCP 模式时在采集器上通过 `telnet` 确认是否能正常访问上述 20033 端口
 - 通信端口为 UDP 模式时在采集器上检查 `trident` 日志中 `Socket send failed`，可能由于网络突然中断导致
 - trident_l4_drop
 - 增大系统-采集器-配置-全景图配置参数-流日志采集速率配置
 - trident_l7_http_dns_drop
 - 增大系统-采集器-配置-全景图配置参数-应用层日志采集速率配置
 - trident_queue_overwritten
 - 以下几个在采集器修改配置文件 `/etc/trident.yaml`，增大 `flow-queue-size` 配置
 - `1-tagged-flow-to-quadruple-generator`
 - `1-tagged-sflow-to-quadruple-generator`
 - `1-tagged-netflow-to-quadruple-generator`
 - 以下几个仅在专属服务器类型（ANALYZER 模式）时出现，采集器修改配置文件 `/etc/trident.yaml`，增大 `analyzer-queue-size` 配置
 - `0.1-bytes-to-parse`
 - `0.2-packet-flowgenerator`
 - `0.3-packet-pipeline`
 - `1-packet-statistics-to-doc`

- 不可配置，长度为 65536
- 1-mini-meta-packet-to-pcap
 - 在采集器修改配置文件/etc/trident.yaml，增大 pcap: queue-size 或 pcap: queue-count 配置；前者表示队列的长度，后者表示 pcap 保存 worker 的并行度
- 以下几个在采集器修改配置文件/etc/trident.yaml，增大 quadruple-queue-size 配置
 - 2-flow-with-meter-to-second-collector
 - 2-flow-with-meter-to-minute-collector
- 2-second-flow-to-minute-aggrer
 - 在采集器修改配置文件/etc/trident.yaml，增大 flow: flow-aggr-queue-size 配置
- 2-doc-to-collector-sender
 - 在采集器修改配置文件/etc/trident.yaml，增大 collector-sender-queue-size 或 collector-sender-queue-count 配置；前者表示队列的长度，后者表示统计发送处理 worker 的并行度
- 3-flow-to-collector-sender
 - 在采集器修改配置文件/etc/trident.yaml，增大 flow-sender-queue-size 或 flow-sender-queue-count 配置；前者表示队列的长度，后者表示原始流发送处理 worker 的并行度

采集器异常

- 告警触发条件：采集器状态异常
- 处理方法：
 - 自检失败：日志所在磁盘剩余空间不足 100MB
 - 自检失败：可用内存不足
 - 分发熔断
 - 分发流量达到限速
 - 到分发点的网关 ARP 无法找到
 - 采集包速率达到限速
 - 到数据节点的网关 ARP 无法找到
 - 分配关联数据节点失败
 - 采集器未注册
 - 采集器进程数超限
 - 采集器线程数超限

自检失败：日志所在磁盘剩余空间不足 100MB

确认日志所在磁盘空间是否充足，若默认日志目录如法扩容，可修改 `trident.yaml` 中的 `log-level`。

自检失败：可用内存不足

使用 `free -m` 确认 Trident 运行环境的内存是否充足，若内存充足可再使用 `cat /proc/meminfo | grep MemAvailable` 确认可用内存是否充足。

如果出现 `free` 显示的内存充足但 `MemAvailable` 不够的情况，检查 `sysctl -A 2>&1 | grep min_free_kbytes` 值是否设置过大，可考虑缩小该值。

分发熔断

前往 Trident 运行环境，使用 `ip r get $分发点 IP` 确认分发的出接口，并查看该网口的流量是否过大，根据情况适当调高熔断阈值。

分发流量达到限速

前往系统-采集器-详情页查看分发流量速率曲线`，通过调高分发流量限速或精细化分发策略解除告警。

到分发点的网关 ARP 无法找到

前往 Trident 运行环境，使用 `arping` 命令确认是否能获取到网关的 ARP。

采集包速率达到限速

前往系统-采集器-详情页查看采集流量速率曲线`，通过调高采集流量限速解除告警。

到数据节点的网关 ARP 无法找到

前往 Trident 运行环境，使用 `arping` 命令确认是否能获取到网关的 ARP。

分配关联数据节点失败

- 前往系统-控制器-列表页，检查采集器所在可用区是否有从控制器，且状态正常。如果没有的话，需要调整从控制器对应的区域和可用区，使得采集器所在可用区有从控制器。
- 前往系统-数据节点-列表页，检查采集器所在可用区的数据节点的关联采集器数量，是否已经达到最大值。如果达到最大值，需要更新授权或者在可用区中增加数据节点。

采集器未注册

前往系统-采集器-列表页，点击注册即可。

采集器 CPU 超限

- 告警触发条件：采集器 CPU 消耗 5 分钟持续达到限制的 70%
- 计算方法：CPU 限制见系统-采集器-配置
- 处理方法：此告警极少触发，若流量过大需要视情况增大 CPU 限制

采集器内存超限

- 告警触发条件：采集器内存消耗 5 分钟持续达到限制的 70%
- 计算方法：内存限制见系统-采集器-配置
- 处理方法：查看系统-采集器-列表点击采集器名进入详情页，根据内存消耗曲线和流量曲线适当调高内存限制

进程启动

- 告警触发条件：采集器进程启动
- 处理方法：
 - 服务器可登录：查看服务器上/var/log/trident/trident.log 日志文件，搜索 WARN 或 ERR 确定重启原因。
 - 不可登录操作系统的采集器：确认系统-采集器-配置中的日志发送是否打开，若已打开可通过若可通过页面上的系统-采集器-列表中点击某个采集器查看日志。

数据节点告警

数据节点丢包

上报模

块	队列名	说明	队列长度配置	队列数量配置
droplet	1-recv-unmarshall	解析采集器遥测数据队列	unmarshall-queue-size	unmarshall-queue-count
droplet	1-receive-to-decode-l4	解析采集器流量日志数据队列	decoder-queue-size	decoder-queue-count
droplet	1-receive-to-decode-l7	解析采集器 HTTP、DNS 流量日志数据队列	decoder-queue-size	decoder-queue-count
droplet	vtap-xxx	遥测数据写 TSDB 队列	metrics-ck-writer: queue-size	metrics-ck-writer: queue-count
droplet	flow_log-xxx	流量数据写 TSDB 队列	flowlog-ck-writer: queue-size	flowlog-ck-writer: queue-count

droplet	analyzer_l4_drop	流量日志采样 丢弃	l4-throttle	decoder- queue-count
droplet	analyzer_l7_http_drop	HTTP 流量日 志采样丢弃	l7-http-throttle	decoder- queue-count
droplet	analyzer_l7_dns_drop	DNS 流量日志 采样丢弃	l7-dns-throttle	decoder- queue-count

- 告警触发条件：数据节点进程内部队列丢包
- 处理方法：
 - 点击告警事件中的数据节点主机名，或访问系统-数据节点列表页面并点击数据节点名字
 - 查看数据节点最近 24 小时系统 CPU、内存、磁盘负载曲线和丢包曲线
 - 判断是否由于系统负载高导致的丢包
 - 若是则需要进行数据节点扩容
 - 查看 <http://<DeepFlow>/grafana-server/> 2.2 Droplet Queue 相应队列负载
 - 若是队列处理能力不足，可以通过修改配置文件增加相应队列长度或队列数量解决
 - 配置文件位置：
`/usr/local/deepflow/templates/droplet/templates/configmap.yaml`
 - 修改配置后需执行 `/usr/local/deepflow/bin/deepflow-deploy -u` 和 `kubectl rollout restart ds/droplet-daemonset -n deepflow` 生效。
 - 否则，查看同时是否有其他告警事件，解决其他告警即可

数据节点负载高

- 告警触发条件：数据节点负载 5 分钟持续达到限制的 70%
- 计算方法：CentOS 系统中通过 `top` 命令查询到的最近 1 分钟 load 值与服务器 CPU 逻辑核数的比值
- 处理方法：
 - 点击告警事件中的数据节点主机名，或访问系统-数据节点列表页面并点击数据节点名字
 - 查看数据节点最近 24 小时系统负载曲线
 - 若超过 5% 的时间达到服务器 CPU 逻辑核数量的 70%，则需要进行服务器扩容
 - 否则，查看同时是否有其他告警事件，解决其他告警即可

数据节点失联

- 告警触发条件：主控制器无法通过 RESTful API 连通故障控制器

- 计算方法：
 - 主控制周期调用 `http://{host}:20014/v1/health/` 探测，若收到回复则认为正常
 - 超过一分钟从未收到回复信息，则认为异常
 - 主控制器每隔 3 秒扫描一次所有控制器和数据节点
- 处理方法：
 - 运维人员收到告警后应当尽快通过带外的方式登录数据节点进行恢复

注意： 数据节点失联时，采集器自动切换发送数据的数据节点，默认切换时间为 1 分钟（由系统-采集器-配置页面的最长同步间隔决定）。最坏情况下可能丢失 4 分钟的故障数据节点负责接收的采集器数据。

注意： 若有控制器或数据节点失联，控制器探测的时间间隔可能由于等待超时大于 3 秒。

建议： 默认情况下通过 22 端口探测 SSH 服务，若部署环境使用其他 SSH 端口需要修改 `/etc/manager.yaml` 中的 `check_port` 并重启 `manager` 进程。

数据节点磁盘空间不足

- 告警触发条件：数据节点任何一块磁盘使用空间大于 70%
- 处理方法：
 - 点击告警事件中的数据节点主机名，或访问系统-数据节点列表页面并点击数据节点名字，查看数据节点最近 24 小时磁盘用量曲线，评估是否需要进行磁盘或服务器扩容
 - 访问系统-数据节点下的存储配置页，查看不同功能对磁盘的消耗，评估是否需要减少功能数据的保留时长

TSDB 写入失败

上报模块	统计项	说明
droplet	ckwriter_metrics_1s	秒级遥测数据写 TSDB 失败次数
droplet	ckwriter_metrics_1m	分钟级遥测数据写 TSDB 失败次数
droplet	ckwriter_flow_log	流量日志数据写 TSDB 失败次数

- 告警触发条件：数据节点 `droplet` 写入 TSDB 时失败
- 处理方法：
 - 登录告警事件中的数据节点主机
 - 查看 TSDB 服务是否正常: 执行 `kubectl -n deepflow get pod | grep clickhouse` 看服务状态是否正常
 - 若服务状态异常，请联系 DeepFlow 技术支持人员。
 - 查看是否有足够的磁盘剩余空间: 执行 `df -h` 看是否磁盘空间占满导致写入失败。

- 若磁盘剩余空间不足，清理磁盘空间或扩容磁盘。

其他告警

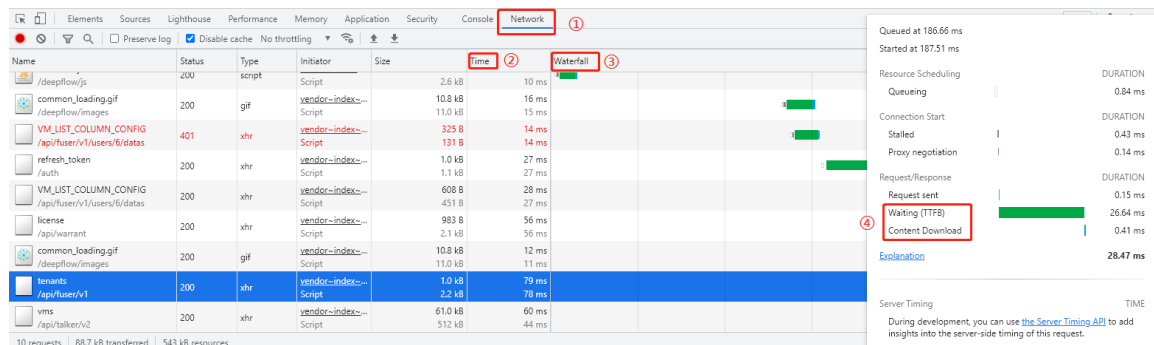
云平台同步异常

- 告警触发条件
 - 云平台 API 或者 SDK 报错
 - K8s 集群中没有部署采集器
- 处理方法：
 - 资源/资源池/云平台列表中，可以看到具体异常
 - 登陆控制器上查看/var/log/cloud-agent.log，搜索 WARN 或 ERR 查看详细日志

慢查询

页面加载慢

页面加载慢时，可在浏览器上按 **F12** 按钮唤出调试页面，下图以 Chrome 为例说明排查方法：



浏览器查看

- 查看 **Network** 标签可排查页面慢的问题
- 观察 **Time** 列，是否有耗时很久的 API
- 也可观察 **Waterfall** 列，直观的看到耗时久的 API，水平绿柱（或蓝柱）越长则耗时越久
- 鼠标悬停瀑布图的某个水平柱上，可显示详情，一般关注 **Waiting** 和 **Content Download** 两项
 - 前者一般可表示服务端 API 耗时
 - 后者表示从 API 首个回复包到最后一个包的时间，通常可结合表格中的 **Size** 列，内容越大传输越耗时

一般可将上图截图给后端支持人员，注意也需要包含第一列（API 信息）。

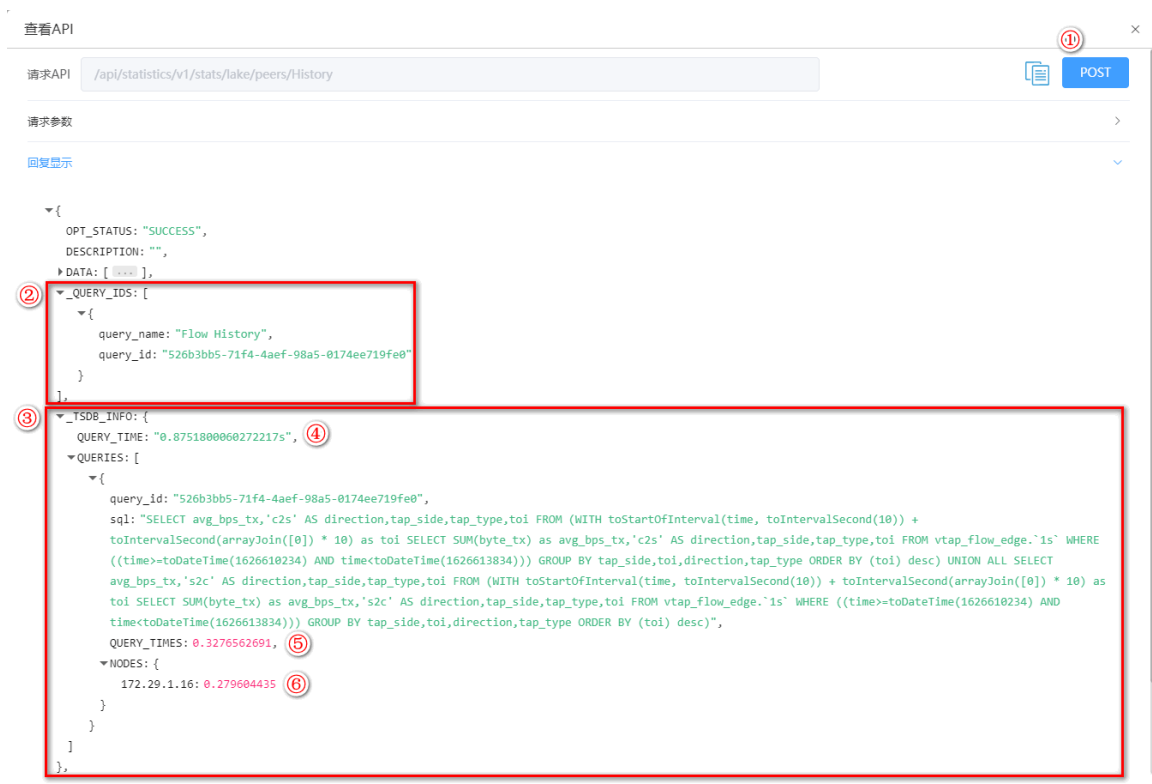
注意：要先打开 F12 调试框，再刷新页面才能查看到 API 调用信息。一般可以在打开 F12 后切换一下菜单，然后点击上图中左上角第二行、第二列的清除图标，然后再切回需要排查问题的菜单来查看。

排查全景图 API 响应慢

对于全景图 API，还可通过子视图的查看 API 功能，或 Kibana APM 功能排查性能瓶颈。全景图流量搜索（除流量日志以外）会依次调用 statistics、clickhouse 进程完成一次页面的 API 请求。

通过 API 返回信息排查

每个子视图右上角的标准操作（齿轮）中，都有查看 API 的操作。点击后弹出框如下图所示：



API-第一步

在弹出页面中点击 POST，可向控制器发送请求，之后点击页面中的回复显示可查看如上图所示的 API 返回信息：- 返回信息中的 _QUERY_IDS 列出了控制器 statistics 调用 clickhouse 的 API 请求，及每个请求对应的含义 - 返回信息中的 _TSDB_INFO 为上述对 clickhouse 的多个请求的详细信息，请求顺序与 _QUERY_IDS 中一一对应 - 红框中的最上方的 QUERY_TIME (4)：表示 statistics 进程响应此次 API 请求的总耗时 - QUERIES 中的每个 QUERY_TIME (5)：表示 statistics 每次查询 clickhouse 的

耗时 - NODES 中每个 IP (6) : IP 表示一个数据节点, 其后的数值代表该数据节点所在区域的 clickhouse 集群查询耗时

一般而言, 可基于上述信息按下面的方法判断性能瓶颈: - 若 (6) 显著大, 说明某个区域的 clickhouse 处理慢 - 若 (5) 显著大, 说明某个 statistics 的子查询慢, 一般是 clickhouse 处理慢 - 若 (4) 显著大, 说明 statistics 处理慢

通过 Kibana APM 排查

除了通过 DeepFlow 页面的查看 API 功能排查单次 API 调用的性能瓶颈以外, statistics 内部使用 OpenTracing 框架进行了性能监控, 可通过 Kibana 页面查看性能瓶颈。

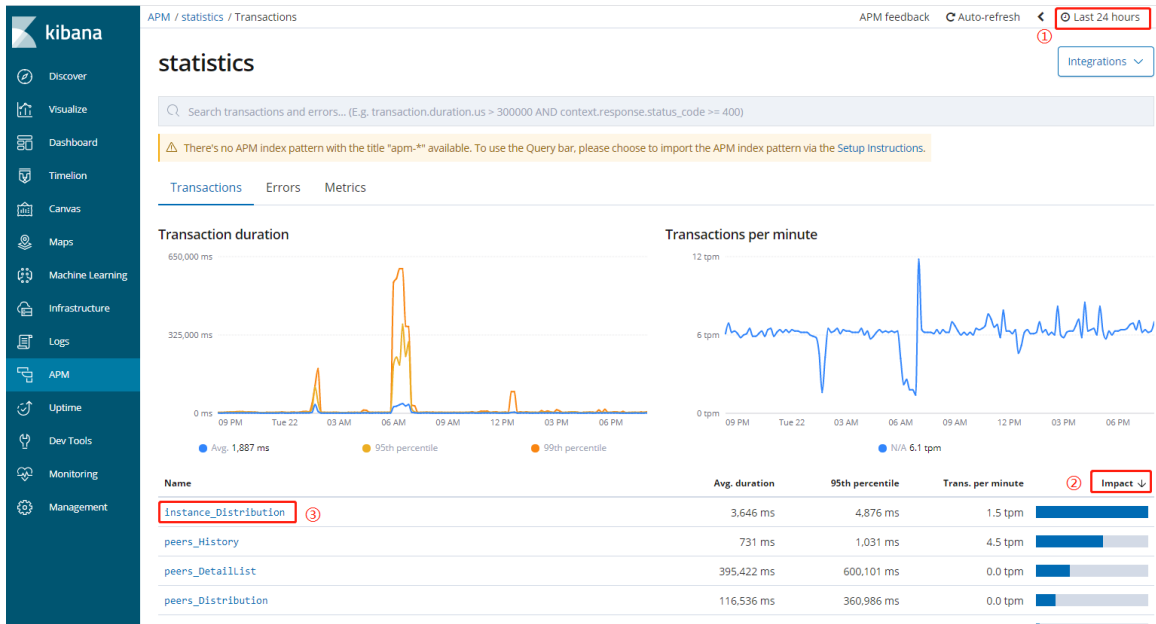
访问方式: 通过 DeepFlow 页面右上角的外部链接中 Kibana 访问, 也可直接访问 [https://\\${DeepFlow 主控制器 IP 或域名}/kibana/](https://${DeepFlow 主控制器 IP 或域名}/kibana/)。页面如下图所示:

Name ↑	Agent	Avg. response time	Trans. per minute	Errors per minute
influxd	go	46 ms	111.0 tpm	0 err.
lake	go	803 ms	13.9 tpm	0 err.
reciter	go	72 ms	111.0 tpm	0 err.
statistics ②	python	1,887 ms	6.1 tpm	0 err.

APM- 第一步

点击上图左侧的 APM, 然后点击右边的 statistics, 可以进入查看 statistics 的应用性能链路追踪数据。

在 statistics 页面中, 可通过右上角时间控件修改时间范围, 页面下方的表格默认按延时从大大小对 API 进行了排序, 点击需要排查的 API 进入特定 API 的监控页面:

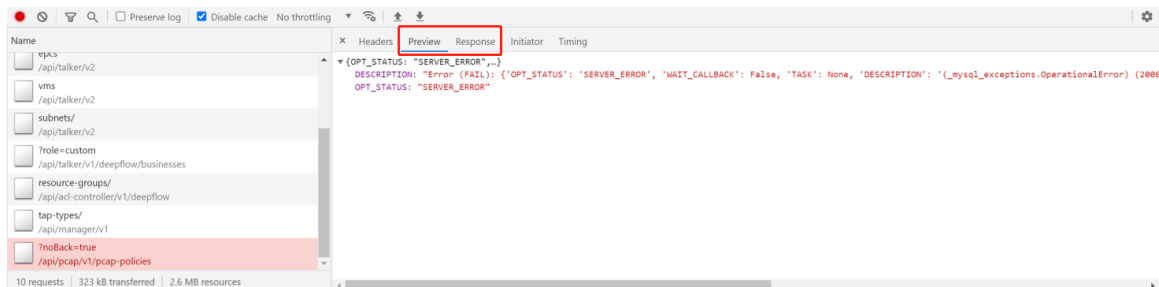


APM- 第二步

在该详情页面中，下图的柱状图表示该 API 的时延分布，可以选择查看具体时延范围的链路追踪图。下方的链路追踪图用不同的颜色标识了不同的函数或服务，并且通过瀑布图的方式展示了 API 调用过程的耗时。直观的来看，某个水平柱状图越长代表对应服务的耗时越大，此时如果它的下方还有耗时很长的柱那么可以沿着它继续寻找瓶颈，否则它本身对应的服务就是性能瓶颈所在。

- 调试框中表格的每一行代表一次页面的 API 调用
- 出错的 API，表格中的第二列 **Status** 一般是 4XX 或 5XX，其中 4XX 一般代表页面下发的请求未通过校验，5XX 一般代表控制器上的服务报错了

点击 **Status** 为 4XX 或 5XX 的某行，会出现该次调用的想起信息，如下图所示：



浏览器查看

一般查看上图中的 **Preview** 或 **Response** 可以拿到详细的出错信息。错误一般显示在 **DESCRIPTION** 等字段中，部分情况下该字段处于折叠状态要将它展开。

最佳实践

推荐配置

最小化部署

参考[软件架构](#)中的最小化部署说明。

性能规格

控制器

- 单控制器最大可管理 **2000** 个采集器
- 控制器集群最大可水平扩展至 **50** 台
- 支持多区域部署，每个区域内可采用 **N+1** 冗余实现高可用

采集器

- KVM、容器
 - 1C1G 采集器：最高可达 10Gbps/300Kpps 采集性能、1Gbps/100Kpps 流量分发性能
 - 2C2G 采集器：最高可达 20Gbps/600Kpps 采集性能、2Gbps/200Kpps 流量分发性能
- ESXi
 - 1C2G 采集器：最高可达 5Gbps/300Kpps 采集性能、1Gbps/100Kpps 流量分发性能

- 4C4G 采集器：最高可达 15Gbps/600Kpps 采集性能、2Gbps/200Kpps 流量分发性能
- 专属服务器
 - 单机标准配置下最高可达 20Gbps/3Mpps 采集性能
- Workload
 - Linux：最高可达 2Gbps 采集性能、1Gbps 流量分发性能

数据节点

- 数据节点集群最大可水平扩展至 1000 台
- 性能参数
 - 指标数据
 - 单台 8CPU 数据节点最高读写速率为 200 万数据点/秒
 - 单台 40CPU 数据节点最高读写速率为 1000 万数据点/秒
 - 若开启跨数据节点数据冗余，数据节点的资源配比需要 x2
 - 流日志数据
 - 单台 8CPU 数据节点最高可写入 2 万条流日志/秒
 - 单台 40CPU 数据节点最高可写入 10 万条流日志/秒
 - 可根据数据节点 CPU/内存/磁盘性能消耗合理调整写入速率限制，超过限制时自动采样写入
 - 若开启跨数据节点数据冗余，需将写入速率限制减半
 - PCAP 数据
 - 单台 8CPU 数据节点最高可写入 200Mbps 数据包
 - 单台 40CPU 数据节点最高可写入 1Gbps 数据包
 - 不提供跨数据节点数据冗余能力，使用物理机 RAID1 或虚拟机高可用云硬盘实现数据冗余存储
 - 仅当用户配置 PCAP 策略时存储对应的 PCAP 数据
- 容量估计
 - 关闭跨数据节点数据冗余时，一个 40CPU 逻辑核的数据节点预估最大可存储 2000 个被监控 IP 地址的指标数据
 - 最高约对应 200 个 A 类（KVM/ESXi/Hyper-V/容器）采集器，或 2000 个 B 类采集器
 - 最高约对应 2 个专属服务器采集器，即平均存储 10Gbps 流量的数据需要 10CPU 逻辑核的数据节点
 - 当部署 C 类全栈采集器时，最高约对应 50 个宿主机上的所有采集器
 - 提示：默认建议关闭跨数据节点数据冗余
 - 物理机数据节点可通过 RAID1 提供本机数据冗余能力，虚拟机数据节点使用云硬盘一般已实现了副本存储

- 若开启跨数据节点数据冗余，数据节点的资源配比需要 x2

资源规格

控制器

项目	最低要求	说明
CPU	8 核	逻辑 CPU / 超线程数量
内存	64G	
系统磁盘	200Gx2	物理机 2 块磁盘 RAID1（虚拟机不需要）
数据磁盘	400Gx2	物理机 2 块磁盘 RAID1（虚拟机不需要），存储系统监控数据（InfluxDB）、告警事件/资源变更事件/系统日志数据（ES）
Web 访问网口	1 个千兆	打开控制器高可用时需配置虚 IP
控制面网口	1 个万兆	参考下方带宽预估章节

采集器

KVM/K8s

在 KVM、K8S 节点环境下，Trident 进程形态资源最低要求为：

项目	最低要求	说明
CPU	1 逻辑核	
内存	1G	
磁盘	100M	/var/log 目录所在分区空闲空间
网络		采集：和控制器 IP 通信 分发：和分发点（隧道对端 IP）通信 监控诊断：和数据节点 IP 通信
操作系统		CentOS 7/8, Kernel >= 3.0 Ubuntu 16/18, Kernel >= 3.0 Debian 10, Kernel >= 3.0

虚拟交换机	Open vSwitch, tap 接口 MAC 地址后 5 字节需与虚拟机 MAC 地址相同
	Linux Bridge, tap 接口 MAC 地址后 5 字节需与虚拟机 MAC 地址相同
	K8S Flannel
	K8S Calico

ESXi

在 vSphere 环境，运行采集器的虚拟机资源最低要求为：

项目	最低要求	说明
CPU	1 逻辑核	
内存	2G	
磁盘	30G 系统盘	
网络		采集：和控制器 IP 通信 分发：和分发点（隧道对端 IP）通信 监控诊断：和数据节点 IP 通信
虚拟交换机		vSphere VDS vSphere VSS

注：操作系统本身会消耗一部分 CPU 和内存资源，上述最低规格下的采集器性能约为 KVM 环境的 70%。

专属服务器

项目	最低要求	说明
CPU	16 核	逻辑 CPU / 超线程数量
内存	64G	
系统磁盘	200Gx2	2 块磁盘 RAID1
数据磁盘	800Gx2	2 块磁盘 RAID1，存储 PCAP 文件
控制面网口	1 个千兆	同时还用于接收 NetFlow、sFlow
数据面网口	4 个万兆	最大处理 20Gbps/3Mpps 流量

隧道解封装

项目	最低要	说明

	求	
CPU	16 核	逻辑 CPU / 超线程数量
内存	16G	
系统磁盘	200Gx2	2 块磁盘 RAID1
控制面网口	1 个千兆	
数据面网口	2 个万兆	10Gbps/10Mpps，接收分发流量、发送隧道解封装后的流量

Workload

直接运行于客户虚拟机、裸金属服务器中时，Trident 进程形态资源最低要求为：

项目	最低要求	说明
CPU		CPU 消耗与流量成正比，可参考性能测试报告，CPU 过载保护最低可设置 1 逻辑核
内存	256M	
磁盘	100M	/var/log 目录所在分区空闲空间
网络		采集：和控制器 IP 通信 分发：和分发点（隧道对端 IP）通信 监控诊断：和数据节点 IP 通信
操作系统		CentOS 7/8, Kernel >= 3.0 Ubuntu 16/18, Kernel >= 3.0 Debian 10, Kernel >= 3.0 Windows Server 2008R2、2012、2016、2019

数据节点

最小配置

项目	配置	说明
CPU	8 核	逻辑 CPU / 超线程数量
内存	64G	
系统磁盘	200Gx2	物理机 2 块磁盘 RAID1（虚拟机不用 RAID）

数据磁盘 1	400Gx2 SSD	物理机 2 块磁盘 RAID1（虚拟机不用 RAID），存储全景图指标数据、流量日志数据（ClickHouse）
数据磁盘 2	400Gx2	物理机 2 块磁盘 RAID1（虚拟机不用 RAID），存储 PCAP 文件
控制面网口	1 个万兆	参考下方带宽预估章节

注意：数据存储高可用场景下，处理性能最高可能折半。

虚拟机标准配置

项目	配置	说明
CPU	20 核	逻辑 CPU / 超线程数量
内存	128G	
系统磁盘	200Gx1	
数据磁盘 1	800Gx1 SSD	存储全景图指标数据、流量日志数据（ClickHouse）
数据磁盘 2	800Gx1	存储 PCAP 文件
控制面网口	1 个万兆	参考下方带宽预估章节

注意：数据存储高可用场景下，处理性能最高可能折半。

物理机标准配置

项目	配置	说明
CPU	40 核	逻辑 CPU / 超线程数量
内存	256G	
系统磁盘	200Gx2	物理机 2 块磁盘 RAID1
数据磁盘 1	800Gx2 SSD	物理机 2 块磁盘 RAID1，存储全景图指标数据、流量日志数据（ClickHouse）
数据磁盘 2	800Gx2	物理机 2 块磁盘 RAID1，存储 PCAP 文件
控制面网口	1 个万兆	参考下方带宽预估章节

注意：数据存储高可用场景下，处理性能最高可能折半。

集群带宽需求预估

虚拟化采集器与数据节点间

假定流量模型为： - 平均每个宿主机/容器节点的流量为 1Gbps - 平均包长 625~1500 字节（虚拟网络环境大包为主） - 平均每个连接 1020 个包，平均每 1Gbps 流量的连接速率为 4K/s20K/s

控制面网口带宽预估： - 全景图：接收 1K 个采集器的遥测数据大约消耗 100Mbps 带宽，约为业务流量的 0.01% - 流日志：接收 1K 个采集器的流日志大约消耗 4Gbps~20Gbps 带宽，约为业务流量的 0.4%~2%，取决于连接速率 - PCAP：若打开全网 PCAP 策略且设置截断=0，接收 1K 个采集器的压缩包头大约消耗 20Gbps~50Gbps 带宽，约为业务流量的 2%~5%，取决于平均包长

专属服务器采集器与数据节点间

- 若全部打开 PCAP 截断=0 的策略，每处理 20Gbps 镜像流量大约消耗 0.4Gbps~1Gbps 带宽用于发送压缩包头，约为业务流量的 2%~5%，取决于平均包长

控制器与采集器间

- 控制器同时向 1000 个采集器同步全量策略和资源信息大约需消耗 1G 带宽

通信矩阵

控制器与外部通信

客户 端	客户端进 程	服务端	服务端进 程	协 议	端口号	功能
外部	-	主备控制 器	openresty	TCP	80	Web 自动跳转
外部	-	主备控制 器	openresty	TCP	443	Web、API
外部	-	主备控制 器	sshd	TCP	61209	SSH
主备 控制 器	postman	邮件服务 器	-		按实际 情况	发送告警、报表
控制 器	cloud- agent	本区域云 平台控制 器	-	TCP	按实际 情况	平台对接，资源同 步
控制 器	chronyd	外部 NTP 服务	-	UDP	123	K8s 计算节点和外 部 NTP 服务同步

注意：当不同区域的外部 NTP 服务 IP 不相同，若 NTP 服务的 IP 是固定的，可以在不同区域的控制器上配置到同一个域名（例如 `external.ntp-server.com`）的不同解析，实现分区同步。

控制器、数据节点内部通信

仅主区域内部

客户端	客户端进程	服务端	服务端进程	协议	端口号	功能
主备控制器	mysqld	主备控制器	mysqld	TCP	20130/30130	主备控制器 MySQL 同步

注意： - MySQL 在在容器部署模式下，使用 30130 端口进行通信

仅所有区域内部

客户端	客户端进程	服务端	服务端进程	协议	端口号	功能
控制器、数据节点	influxdb-relay	区域主节点	influxdb	TCP	20044	DeepFlow 自身性能监控数据
数据节点	clickhouse	数据节点	clickhouse	TCP	9009,9010	ClickHouse 集群交互
数据节点	droplet	数据节点	clickhouse	TCP	9000,9001	写入指标、流量日志数据
数据节点	droplet	流日志接收方	-	TCP	20204	将流日志分发到外部系统
数据节点	droplet	控制器	trisolaris	TCP	20035	数据节点向控制器请求策略
控制器、数据节点	elasticsearch	控制器、数据节点	elasticsearch	TCP	20041	ES 集群同步

注意，同一个区域内，仅在所有数据节点与设置了参与聚合=是的数据节点之间存在 ClickHouse 的双向交互。通过设置该属性，可以降低防火墙配置工作量，例如：
- 一个区域中有数据节点 A~I 共 9 个 - ABC 服务于 AZ1，DEF 服务于 AZ2，GHI 服务于 AZ3 - 若设置所有 9 个数据节点的参与聚合=是，则 9 个数据节点之间全部存在双向交互 - 若设置 A~C 三个数据节点的参与聚合=是，仅在 ABC 内部、ABC 与 DEF 之间、ABC 与 GHI 之间存在交互，而 DEFGHI 之间不存在交互

跨区域及区域内

客户端	客户端进程	服务	服务端进程	协	端口号	功能

	程	端	程	议		
控制器	manager	控制器	trisolaris	TCP	20014	健康探测
控制器	manager	控制器	influxdb	TCP	20044	查询 DeepFlow 自身性能监控数据
控制器	manager	数据节点	pcap-rest	TCP	20205	健康探测
控制器	statistics	数据节点	clickhouse	TCP	9000,9001	全景图指标数据、流量日志查询
控制器	manager	数据节点	droplet	TCP	20106	数据源配置管理
控制器	pcap	数据节点	pcap-rest	TCP	20205	PCAP 查询和下载
控制器	pcap	控制器	influxdb	TCP	20044	查询 PCAP 文件大小和个数
数据节点	droplet	控制器	trisolaris	TCP	20035	数据节点自动发现
控制器	trisolaris	控制器	mysqld	TCP	30130	控制器 Trisolaris 访问
控制器、区域主节点	genesis	控制器	mysqld	TCP	30130	控制器 Genesis 访问
控制器、区域主节点	grafana-server	控制器	mysqld	TCP	30130	获取 grafana 配置信息
控制器	trisolaris	控制器	manager	TCP	20403	系统管理 API
控制器	trisolaris	控制器	warrant	TCP	20413	证书授权 API
控制器	cloud-agent	控制器	talker	TCP	20013	获取云平台配置信息
控制器	acl-controller	控制器	trisolaris	TCP	20014	刷新策略信息
控制器	talker	控制器	trisolaris	TCP	20014	刷新分发点信息
控制器	manager	控制	trisolaris	TCP	20014	刷新采集器信息

控制器	talker	器 控制 器	cloud-agent	TCP	20230	下发云平台配置信息
控制器	exchange	控制 器	cloud-agent	TCP	20230	获取云平台数据

注意： - MySQL 在容器部署模式下，使用 30130 端口进行通信

采集器通信

采集器所在区域的控制器

客 户 端	客户端 进程	服 务 端	服务端进 程	协 议	端口号	功能
采 集 器	trident	控 制 器	droplet	UDP	20033	采集器发送自身监控数据和日志到关联控制器上，监控数据最终存储于主备控制器 InfluxDB 中
采 集 器	trident	控 制 器	trisolaris	TCP	20035, 20135	采集器向控制器 trisolaris 请求策略、远程升级、请求 chronyd 时间、请求 Genesis 资源同步。二选一，明文/SSL 通信
控 制 器	trident-ctl	采 集 器	trident	UDP	20035	采集器向控制器发送信标以便于反向使用 trident-ctl 获取 debug 信息和 profile

上述 trident-ctl、go、wget、curl 可在控制器上安装，用于对采集器的功能和性能问题进行排查。

主区域的控制器

客 户 端	客户端 进程	服 务 端	服务端进 程	协 议	端口 号	功能
控 制 器	pcap	采 集 器	pcap-rest	TCP	20205	仅专属服务器类型的采集器，提供 PCAP 下载 API

采集器所在区域的数据节点

客 户 端	客户端 进程	服 务 端	服务端进 程	协 议	端口 号	功能
采 集 器	trident	数 据 节	droplet	TCP/UDP	20033	发送指标数据、流量日志、PCAP，用于全景图流量搜索、流量日志、流量下载。

点

TCP/UDP 二选一

分发点

客户端	客户端进程	服务端	服务端进程	协议	端口号	功能
采集器	trident	分发点	-	UDP	4789	包分发, VXLAN 隧道
采集器	trident	分发点	-	GRE	-	包分发, ERSPAN 隧道
采集器	trident	采集器	river, rosen	UDP	4789	仅隧道卸载模式的采集器, 用于接收分发流量、卸载 VXLAN

仅发生在一个 K8s 集群内部的通信

节点类型	服务端进程	协议	端口号	功能
控制器	cerebro	TCP	20052	ES 监控, 通过 openresty 代理到 443
控制器	cloud-agent	TCP	20230	云平台资源同步
控制器	fauth	TCP	20822	授权管理
控制器	fpermit	TCP	20823	权限认证
控制器	fuser	TCP	20824	用户管理
控制器	df-web-service	TCP	20825	
控制器	kibana	TCP	5601	ES 查询 GUI
控制器	rabbitmq	TCP	4369	
控制器	rabbitmq	TCP	20020	
控制器	rabbitmq	TCP	20001	
控制器	redis	TCP	6379	
控制器	talker	TCP	20013	云平台资源 API
控制器	acl-controller	TCP	20408	策略 API
控制器	statistics	TCP	20404	全景图流量搜索 API
控制器、数据节点	influxdb	TCP	20044	DeepFlow 自身性能监控数据

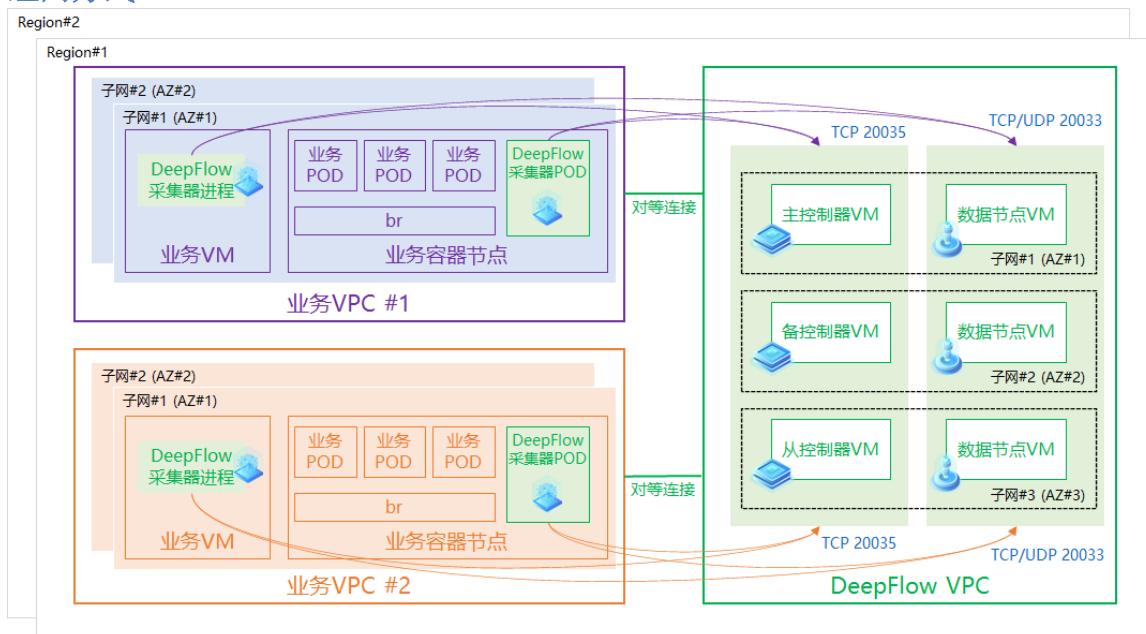
点					
控制器、区域主节点	grafana-server	TCP	20060		展示 DeepFlow 性能监控数据
控制器	report	TCP	20400		报表 API
控制器	alarm	TCP	20401		告警 API
控制器	diagnose	TCP	20407		全景图虚拟网络诊断 API
控制器	pcap	TCP	20409		PCAP 下载 API
控制器	warrant	TCP	20413		授权 API
控制器	monitor	TCP	20415		记录资源事件，提供资源事件 API
数据节点	droplet	TCP	9526		性能故障排查，go profile
数据节点	clickhouse	TCP	8123,8124		自身性能监控，供 telegraf 读取
控制器、数据节点	influxdb-relay	TCP	20048		收集 DeepFlow 自身性能监控数据

公有云最佳实践

部署形态

DeepFlow 在公有云（特指客户的专享部署方式，非 SaaS）中全部使用虚拟机部署控制器和数据节点。特别地，若客户有高性能隧道解封装需求，可使用公有云上的裸金属服务器部署隧道解封装采集器。

组网方式



公有云组网

- 控制器、数据节点：DeepFlow 控制器和数据节点部署于单独的 DeepFlow VPC 中，通过对等连接为业务 VPC 提供监控服务
 - 对等连接可配置 ACL，最小仅需放行到控制器的 TCP 20035 及到数据节点的 TCP/UDP 20033 端口
 - 若由于 CIDR 冲突无法规划一个单独的 VPC 与所有业务 VPC 对等连接打通，需创建多个 DeepFlow VPC 分别服务于一部分业务 VPC
- 跨区域：不同区域中的业务 VPC，请求各自区域中的 DeepFlow 控制器和数据节点，并挑选其中一个区域作为主区域部署主控制器
- 高可用：公有云虚拟机、云硬盘有高可用能力，如非特殊需要无需部署备控制器、无需设置数据节点数据冗余
 - 若希望提供主备控制器自动切换，请在主区域中选择三个可用区（三个子网）分别部署主备从控制器
 - 若希望提供数据节点冗余数据存储（不建议），请将同一个分片（shard）的两个数据节点部署在不同的可用区（子网）中

资源配置

资源标准配置为：- 控制器：虚拟机，请参考控制器标准配置，但注意磁盘无需配置 RAID 1、网口无需限定万兆 - 数据节点：虚拟机，请参考数据节点虚拟机标准配置，但注意磁盘无需配置 RAID 1、网口无需限定万兆 - 专属服务器采集器：不提供此能力 - 隧道解封装采集器：虚拟机/裸机，请参考隧道解封装采集器标准配置，但注意磁盘无需配置 RAID 1、网口无需限定万兆。

公有云虚拟机可弹性扩缩容，对于小规模部署可适当选用低规格（不能低于 8C16G）虚拟机起步，磁盘、网卡按需配置按需扩容。

数据节点的数据存储未来将支持通过对象存储服务，将冷数据自动迁移至更经济、更稳定的存储介质中。

扩容方式

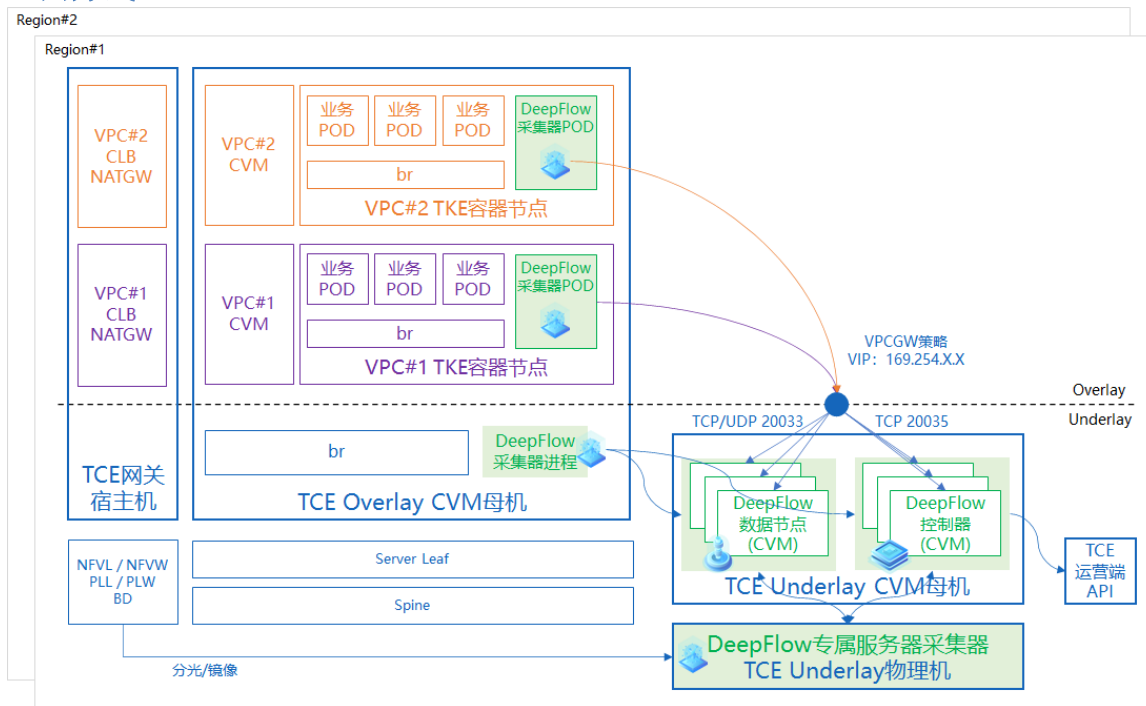
各组件所在操作系统 24 小时峰值负载超过 CPU 核数的 70%就需进行扩容。在单节点资源未达到标准配置前，请通过垂直扩展的方式扩容当前节点。当达到标准配置之后，请通过水平扩展的方式扩容集群。

腾讯 TCE 最佳实践

部署形态

DeepFlow 在腾讯专有云（TCE）中使用 Underlay CVM 部署控制器和数据节点，使用 TCE 标准服务器部署专属服务器采集器和隧道解封装采集器。

组网方式



腾讯 TCE 组网

- 控制器、数据节点：DeepFlow 控制器和数据节点以 CVM 形态部署于 Underlay 中
 - KVM（母机）上的采集器可直接与之进行通信

- 租户 TKE 容器集群中的采集器可通过配置全局 VPCGW 策略与 Underlay 中的控制器与数据节点集群通信
- 控制器需要请求运营端 API 服务
- 跨区域：不同区域中的采集器请求各自区域中的 DeepFlow 控制器和数据节点，并挑选其中一个区域作为主区域部署主控制器
- 高可用：Underlay CVM 有高可用能力，如非特殊需要无需部署备控制器、无需设置数据节点数据冗余
 - 若希望提供主备控制器自动切换，请在主区域的 Underlay 中选择三个可用区（三个子网）分别部署主备从控制器
 - 若希望提供数据节点冗余数据存储（不建议），请将同一个分片（shard）的两个数据节点部署在不同的 Underlay 可用区（子网）中

资源配置

资源标准配置为：- 控制器：Underlay CVM，请参考控制器标准配置，但注意磁盘无需配置 RAID 1、网口无需限定万兆 - 数据节点：Underlay CVM，请参考数据节点虚拟机标准配置，但注意磁盘无需配置 RAID 1、网口无需限定万兆 - 专属服务器采集器：TCE 标准服务器，最低配置请参考专属服务器采集器标准配置 - 隧道解封采集器：TCE 标准服务器，最低配置请参考隧道解封采集器标准配置

数据节点的数据存储未来将支持通过对象存储服务，将冷数据自动迁移至更经济、更稳定的存储介质中。

扩容方式

各组件所在操作系统 24 小时峰值负载超过 CPU 核数的 70% 就需进行扩容，所有组件通过水平扩展的方式扩容集群。

在 TCE 页面的配置

详细内容请参考[云杉内部文档](#)，包括如下内容：- TKE 容器节点请求 Underlay 中的控制器和数据节点 - DeepFlow 容器采集器自动升级 - DeepFlow Web 页面访问 - 在 TCE 页面增加 DeepFlow Web 页面跳转链接

阿里专有云最佳实践

部署形态

DeepFlow 在阿里专有云（Apsara Stack）中目前有两种部署方式：- 请参考阿里云已发布的[混合云网络流量分析三方联合最佳实践](#)，以物理机形态部署控制器和数据节点 - 使用 NFV ECS 形态部署控制器和数据节点，相关文档随阿里专有云新版本发布同步提供

组网方式

请参考阿里云在线最佳实践文档。

资源配比

若以物理机部署 DeepFlow，资源标准配置为： - 控制器：阿里专有云标准服务器，最低配置请参考控制器标准配置 - 数据节点：阿里专有云标准服务器，最低配置请参考数据节点物理机标准配置 - 专属服务器采集器：阿里专有云标准服务器，最低配置请参考专属服务器采集器标准配置 - 隧道解封装采集器：阿里专有云标准服务器，最低配置请参考隧道解封装采集器标准配置

若以 NFV ECS 部署 DeepFlow，资源标准配置为： - 控制器：NFV ECS，请参考控制器标准配置，但注意磁盘无需配置 RAID 1、网口无需限定万兆 - 数据节点：NFV ECS，请参考数据节点虚拟机标准配置，但注意磁盘无需配置 RAID 1、网口无需限定万兆 - 专属服务器采集器：暂不提供此能力 - 隧道解封装采集器：NFV ECS，请参考隧道解封装采集器标准配置，但注意磁盘无需配置 RAID 1、网口无需限定万兆

数据节点的数据存储未来将支持通过对象存储服务，将冷数据自动迁移至更经济、更稳定的存储介质中。

扩容方式

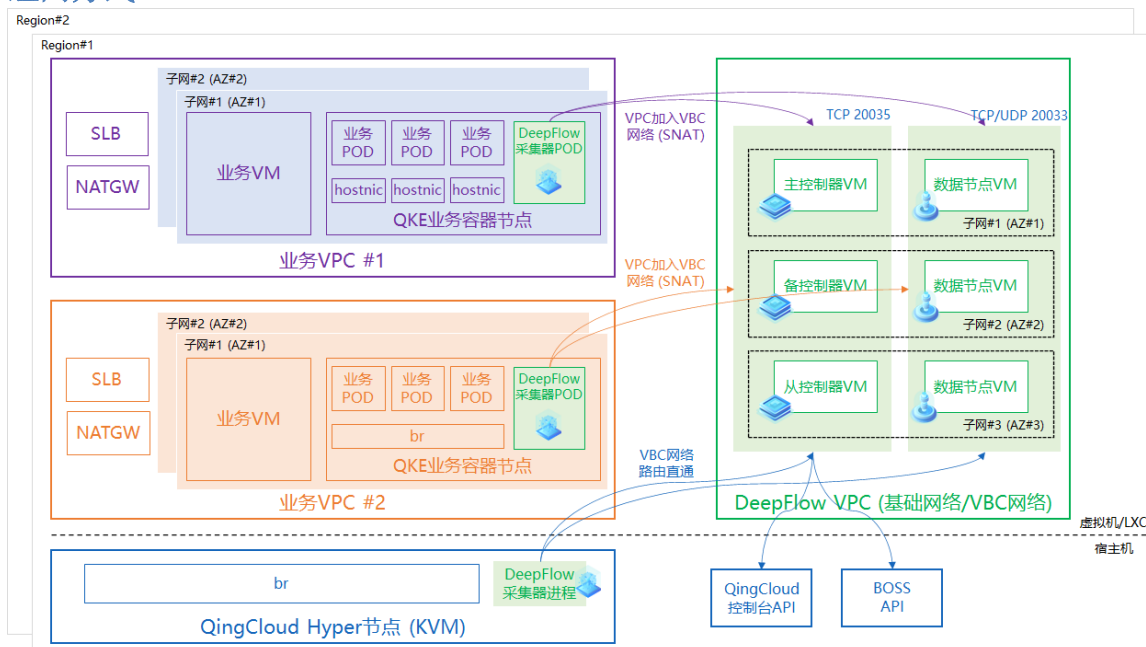
各组件所在操作系统 24 小时峰值负载超过 CPU 核数的 70%就需进行扩容，所有组件通过水平扩展的方式扩容集群。

青云私有云最佳实践

部署形态

DeepFlow 在青云私有云（QingCloud）中使用基础网络（VBC 网络）中的虚拟机部署控制器和数据节点，使用青云私有云标准服务器部署专属服务器采集器和隧道解封装采集器。

组网方式



青云私有云组网

- 控制器、数据节点：DeepFlow 控制器和数据节点以虚拟机形态部署于基础网络（VBC 网络）中
 - KVM（Hyper 节点）上的采集器可直接与之进行通信
 - 租户 QKE 容器集群中的采集器，需将所在 VPC 加入 VBC 网络，通过 SNAT 访问访问控制器与数据节点集群
 - 控制器需要请求 QingCloud 控制台 API 服务、BOSS API 服务
- 跨区域：不同区域中的采集器请求各自区域中的 DeepFlow 控制器和数据节点，并挑选其中一个区域作为主区域部署主控制器
- 高可用：虚拟机有高可用能力，如非特殊需要无需部署备控制器、无需设置数据节点数据冗余
 - 若希望提供主备控制器自动切换，请在主区域中选择三个可用区（三个 VBC 子网）分别部署主备从控制器
 - 若希望提供数据节点冗余数据存储（不建议），请将同一个分片（shard）的两个数据节点部署在不同的可用区（VBC 子网）中

资源配置

资源标准配置为：- 控制器：虚拟机，请参考控制器标准配置，但注意磁盘无需配置 RAID 1、网口无需限定万兆 - 数据节点：虚拟机，请参考数据节点虚拟机标准配置，但注意磁盘无需配置 RAID 1、网口无需限定万兆 - 专属服务器采集器：青云私有云标准服务器，最低配置请参考专属服务器采集器标准配置 - 隧道解封封装采集器：青云私有云标准服务器，最低配置请参考隧道解封封装采集器标准配置

数据节点的数据存储未来将支持通过对象存储服务，将冷数据自动迁移至更经济、更稳定的存储介质中。

扩容方式

各组件所在操作系统 24 小时峰值负载超过 CPU 核数的 70%就需进行扩容，所有组件通过水平扩展的方式扩容集群。

部署扩容

控制器与数据节点部署

本章节是对 DeepFlow 控制器、数据节点的部署方法说明。自 5.7.3 之后版本，均采用 K8s 部署模式。

准备工作

DeepFlow 软件包和授权

DeepFlow 容器化部署软件包获取

DeepFlow 容器化部署软件包为 iso 格式，deepflow-docker-release*.iso 内含离线部署镜像仓库，trident 安装包，容器化部署工具包等离线部署所需内容。

DeepFlow 授权文件申请步骤

安装完成后，如选择的为软件授权模式，DeepFlow 各种功能需要授权，才可以正常使用，授权文件申请是不可或缺的一步，以下为申请授权文件步骤：

- 申请软件授权，需要提供：控制器 IP、控制器主机名、授权天数、数据节点 CPU 核数、不同类型的采集器数量、客户名称相关信息
 - 目前采集器分为：KVM、ESXi、容器-V、容器-P、专属服务器、Workload-V、Workload-P 几种
- 授权文件以 DEEPFLOW-LICENSE-xxxxxx-xxx-xxxx.LIC 命名，例如：DEEPFLOW-LICENSE-20191009-160833-1003.LIC，
- 得到授权文件后，访问主控制器的 443 端口,会出现 DeepFlow 登陆界面，输入用户名密码，在主界面上点击上传授权文件按钮即可上传
- 上传成功后，会出现 DeepFlow 等相关内容，确认无误后，所有功能授权成功
- 申请证书时，控制器主机名和控制器 IP 地址只填写主备控制器的名称和 IP

DeepFlow 代金券申请步骤

安装完成后，如选择的按量计费模式，DeepFlow 的功能，需要充值代金券才可以正常使用，代金券申请是不可或缺的一步，以下为申请代金券步骤：

- 申请代金券，需要提供：
 - 客户名称
 - 合同编号
 - CRM_ID
 - DeepFlow 版本
 - 控制器主机名：任意控制器主机名都可
 - 控制器 IP 地址：任意控制器主机名都可
- 代金券以 DEEPFLOW-yyMMDD –四位随机码-\$CRMID.CPN 命名，例如：DEEPFLOW-210403-xysz-987.CPN
- 得到代金券后，登录 DeepFlow 界面，点击系统-费用中心-充值，上传代金券
- 上传成功后，确认费用中心的可用余额，余额无误则可正常使用 DeepFlow 平台

DeepFlow 服务器初始化

硬件服务器 BIOS/BMC/RAID 配置

- BIOS 和 BMC 配置
 - 设置引导方式为传统 BIOS 而非 UEFI, 否则显示器出现黑屏
 - 设置日期、时间、IPMI IP 地址
 - 修改 BMC 管理账户密码
- 关闭服务器节能模式
 - 服务器开机进入 BIOS 设置：选择 System Setup -> System BIOS -> System Profile Settings
 - System Profile 项目选择 Performance 然后保存退出再次启动即可
- RAID 配置
 - 系统盘
 - 所有服务器的两块 200G SSD 配置 RAID 1 作为系统盘
 - 数据盘
 - 存储 PCAP 文件的磁盘建议配置 RAID 1 冗余，PCAP 无跨服务器备份机制
 - 存储 TSDB 数据的磁盘不强制配置 RAID 冗余，TSDB 支持集群和数据副本的能力
 - 存储流日志、系统日志、告警事件的磁盘不强制配置 RAID 冗余，Elasticsearch 支持集群和数据副本的能力

服务器操作系统安装

所有服务器均需使用 CentOS-7-x86_64-Minimal-2009.iso 安装操作系统，且安装为 Minimal 模式。

如果控制器有公网接口，需要追加如下配置到/etc/sysconfig/network-scripts/ifcfg-xxx:

ARPCHECK=no

警告： - 选择安装为 **Minimal** 且不要安装任何额外软件包，否则 **kubernetes** 将无法部署成功 - 不要对系统盘进行精细分区，保证服务器根分区大小，避免异常状态下系统盘写满。**DeepFlow** 进程日志默认保存至/var/log/目录，当根分区较小，且无法重新分区时，可以通过软链接的方式将该目录迁移至容量充足的分区

建议： 如果操作系统安装在虚拟机上，建议关闭交换空间以避免磁盘 IO 影响虚拟机整体性能；如果产生内存不足的现象，需要加大虚拟机的内存分配。具体操作如下：

```
swapoff -a  
sed -i '/swap/d' /etc/fstab
```

DeepFlow 网络拓扑

DeepFlow 使用到的 IP 地址资源如下所示： - 集群内部控制网络 - 区域内部：每个 DeepFlow 控制器、数据节点需要一个内网 IP 地址，用于互相通信 - 区域之间：主控制器所在区域与其他区域的控制器、数据节点需要一个内网 IP 地址，用于互相通信 - 外部通过 VIP 访问 DeepFlow Web（可选） - DeepFlow 主备控制器需要一个提供外部 Web 访问的 IP 地址，并支持在两台控制器之间进行切换 - 主备控制器公网接口修改为相同的配置，然后将备控制器的公网接口 shutdown

建议： 当数据节点使用虚 IP 接收采集器流量时，所有数据节点控制平面必须要有一个默认网关，确保采集器与数据节点虚 IP 互通。

安装部署 kubernetes

简介

kubeplay 是基于 kubespray 实现的离线部署 kuberneres 集群的工具

特性

- 包含所有依赖，一条命令即可完成离线安装
- 支持 amd64 和 arm64 CPU 架构
- kubeadm 生成的证书有效期调整为 10 年
- 去 docker 化部署，无缝迁移至 containerd 作为容器运行时
- 适用于 toB 私有化场景，可离线安装平台所依赖的 rpm/deb 包（如存储客户端）
- 多集群部署，支持在 kubernetes 集群中以 Job Pod 方式部署 kubernetes 集群
- 使用 GitHub Actions 构建离线安装包，无需充值会员，100% 开源 100% 免费

所有节点安装 Kubernetes 需要的磁盘空间

注：不含 kubeplay 安装包和工作目录

- 系统盘：安装 Kubernetes 大概需要 2G 的磁盘空间，其中/tmp 下临时文件 500M 左右
- 数据盘：建议最小为 Kubernetes 配置 100GB 的数据盘，可以与 DeepFlow 使用同一块数据盘，如果使用系统盘，可能会导致日志等把系统盘磁盘空间打满导致系统服务异常
- Etcd 备份文件保存在系统盘下，备份文件大小和 Kubernetes 的集群规模及数据多少有关，保存最近 5 次的备份文件

compose

在部署工具运行节点使用 `nerdctl compose` 启动 nginx 和 registry 容器，分别提供离线资源下载和镜像分发服务。

kubespray

使用 kubernetes 社区的 `kubespray` 作为集群部署的功能，部署过程中所依赖的资源从 compose 节点获取。

部署

节点规划建议

节点总数小于 3：建议规划一个节点作为 Etcd+Master 节点，其余节点作为 Node 节点。

节点总数大于等于 3：建议规划三个节点作为 Etcd+Master 节点，其余节点作为 Node 节点。

如果客户给的虚拟机分布在不同的可用区，则建议 Etcd 和 Master 节点分布在不同的可用区保证高可用。

升级内核

kubeplay 不会升级内核，CentOS7 需要自行升级内核

必须使用 DeepFlow 团队提供的 Linux 内核

CentOS7 的内核 rpm 包在 ISO 中的 extra-artifacts/kernel-installation-package/ 目录下获取，仅支持 CentOS7，不支持 Redhat7。

如果已经安装了 4.19 内核，先把旧版本删除，然后安装新版本。

```
/usr/bin/rpm -e kernel-4.19.17-1.el7.x86_64 --nodeps  
/usr/bin/rpm -ivh --force ./kernel-4.19.17-1.el7.x86_64.rpm --nodeps
```

登录所有节点并重启

```
shutdown -r now
```

等服务器重启完成后，使用 `uname` 检查是否已使用新的 Linux Kernel。

```
uname -r
```

4.19.17-1.el7.x86_64

检查所有节点的 DNS 配置

检查所有 Kubernetes 节点操作系统的 DNS 配置文件，确保 DNS 服务器不能为空，不能为本机 IP、localhost、127.0.0.1 等，必须配置一个可以用的 IPV4 地址 DNS 地址，比如网关等，注意 DNS server 数量不要超过三个，把正确的地址放到前面。

各系统的 DNS 配置文件： - CentOS 系统 `/etc/resolv.conf` - Ubuntu 系统：

`/run/systemd/resolve/resolv.conf` - Fedora 系统：

`/run/systemd/resolve/resolv.conf`

检查默认的 kubernetes 集群的 pod 网段和 service ClusterIP 网段是否和物理网络等有冲突

kubernetes 集群的 pod 网段和 service ClusterIP 默认网段如下

- kube_service_addresses: 10.233.0.0/18
- kube_pods_subnet: 10.233.64.0/18

如果有冲突可以通过修改 `config.yaml` 文件中的如下配置项调整 kubernetes 集群的网段配置

（如果 `config-sample.yaml` 文件中没有这个配置项，直接复制进去即可，注意层级关系在 `kubespray` 下面）

kubespray:

k8s 的 service 和 pod 的网段，如果和物理网络等有冲突，可通过修改如下变量配 service 和 pod 的网段

kube_service_addresses: 10.233.0.0/18

kube_pods_subnet: 10.233.64.0/18

如果数据盘用的 xfs 文件系统，需要检查 xfs 文件系统参数

ext4 等其他文件系统无需关注。

如果 kubernetes 使用 containerd，则需要确认用作 kubernetes 数据盘的 xfs 文件系统的参数。安装前确认 xfs 的文件系统 `ftype` 参数是否等于 1，在较老的 centos 版本，例如 7.4 7.5 中，默认格式化的 xfs 文件系统 `ftype` 可能等 0。

以 kubernetes 数据目录在 `/opt` 为例：

```
xfs_info /opt |grep ftype
```

在等于 0 的情况下，如果该磁盘是空的，就可以直接重新格式化，有剩余的磁盘、分区可以用作 `kubernetes` 数据盘时，也可以直接格式化：

```
mkfs.xfs -f -n ftype=1 /dev/xxx
```

如果没有多余的磁盘或者分区，数据量小，可以迁移，那就建议迁移数据，重新格式化磁盘。

如果实在没有多余的磁盘/分区，且无法迁移时，最后的兜底方案为虚拟磁盘（见下面的虚拟磁盘使用方法）。

下载安装包

- 公网下载地址

在阿里云对象存储 `df-patch-no-delete/kubeplay/` 下载最新的 `release` 版本离线安装包

包介绍：以 `Release-02` 版本 `CentOS:7` 为例

包名	大小	用途
kubeplay-release-kubernetes-1.21.5-02.1-centos-7-amd64.tar.gz	1.38 GB	适用于 CentOS7 和 Redhat7 amd64 架构的安装包
kubeplay-release-kubernetes-1.21.5-02.1-centos-7-arm64.tar.gz	1.26 GB	适用于 CentOS7 和 Redhat7 arm64 架构的安装包
kubeplay-release-kubernetes-1.21.5-02.1-centos-7.sha256sum.txt	246 Bytes	安装包 sha256sum 校验文件

- 内网下载地址

`http://10.33.30.40:8080/kubeplay/`

DeepFlow 支持的操作系统：

- CentOS:7.9

除这个系统外的其他系统、其他版本，可能出现不适配、不兼容的问题

解压

特别是 `ubuntu` 系统要注意，不能将安装包放到 `/root` 下，因为 `apt` 安装包是用的 `apt` 用户，对 `/root` 没有权限，会导致无法安装包，所以建议放到 `/opt/`

```
tar -zxvf kubeplay-beta-20-kubernetes-1.21.5-centos-7-amd64.tar.gz -C /opt/
```

`/opt/kubeplay` 为 `kubelay` 的工作目录，里面有所有的容器镜像，此目录不可删除

配置部署参数

```
cd /opt/kubeplay
cp config-sample.yaml config.yaml
vi config.yaml
```

config.yaml 配置文件主要分为如下几个部分:

- compose: nginx 和 registry 部署节点信息
- kubespray: kubespray 部署配置
- invenory: kubernetes 集群节点 ssh 登录信息
- default: 一些默认的参数

compose

参数	说明	示例
internal_ip	部署工具 kubeplay 所在服务器内网 IP	192.168.10.11
nginx_http_port	部署 nginx 服务暴露的端口	8080
imagerepo_domain	部署 registry 镜像仓库服务的域名	kube.registry.local
registry_https_port	部署 registry 镜像仓库服务的端口	8081

compose:

```
# Compose bootstrap node ip, default is Local internal ip
internal_ip: 172.20.0.25 ####kubeplay 部署节点IP
# Nginx http server bind port for download files and packages
nginx_http_port: 8080
# Registry domain for CRI runtime download images
imagerepo_domain: kube.registry.local
registry_https_port: 8081
```

kubespray

参数	说明	示例
kube_version	kubernetes 版本号, 每个安装包只支持一个系统	v1.21.5
external_apiserver_access_ip	列表, 集群 APIserver 外部访问 IP, 例如有公网 IP 等	192.168.10.100
kube_network_plugin	选用 CNI 网络插件名称	calico
flannel_interface_regex	flannel 的网卡正则匹配过滤规则	'172.20.1.*'
container_manager	容器运行时	containerd
etcd_deployment_type	etcd 部署方式	host
etcd_events_cluster_setup	是否启动 event-etcd	true
etcd_events_cluster_enabled	是否启动 event-etcd	true
data_dir	kubernetes、containerd、container log 数据目录路径	/opt

kubespray:

```
# Kubernetes version by default, only support v1.21.5
kube_version: v1.21.5 ##默认即可，不需要修改
#### k8s 的 service 和 pod 的网段，如果和物理网络等有冲突，可通过修改如下变量配 service 和 pod 的网段
kube_service_addresses: 10.233.0.0/18
kube_pods_subnet: 10.233.64.0/18
# For deploy HA cluster you must configure a external apiserver access ip
##自定义 apiserver 证书的扩展 IP 地址和域名，列表
####一般用于该节点有公网 IP、域名等，配置额外公网 IP、域名后，就可以通过配置等 IP、域名直接访问 apiserver，单纯的內网无需修改此配置
external_apiserver_access_ip:
- 127.0.0.1
# CNI support calico, flannel, canal, cilium
# Set network plugin to calico with vxlan mode by default
kube_network_plugin: calico ##没有特殊需求保持默认即可，无需修改
####定义 flannel 的默认网卡，如果 node 节点上的默认路由网卡作为 flannel 的网卡不符合预期，可以手动配置下述网段
##flannel_interface_regex: '172\.20\.1\.*' ##cni 选择 flannel 的时候定义，calico 时无需定义
#Container runtime, only support containerd if offline deploy
container_manager: containerd ##默认即可，无需修改
# Now only support host if use containerd as CRI runtime
etcd_deployment_type: host ##默认即可，无需修改
# Settings for etcd event server
etcd_events_cluster_setup: true ##默认即可，无需修改
etcd_events_cluster_enabled: true ##默认即可，无需修改
#### 这部分定义数据目录路径，如果不需要定义数据目录的路径，可以直接将这几行注释掉（截止到下个注释），删除 data_dir 必须同时删除下面的几个配置
data_dir: /opt
etcd_data_dir: "{{ data_dir }}/lib/etcd"
etcd_events_data_dir: "{{ data_dir }}/lib/etcd-events"
kubelet_data_dir: "{{ data_dir }}/lib/kubelet"
containerd_storage_dir: "{{ data_dir }}/lib/containerd"
####支持删除 pods_log_dir 配置，删除该配置 默认的 pod 日志目录为/var/log/pods，一般建议将 pod 的日志目录放到数据盘下，删除该目录建议用在以下场景中
####- /var/log 目录本身就是数据盘目录到/var/log 的软连接，也就是本身/var/log 目录已经为升级盘了，该场景可以删掉此配置避免软连接出问题
pods_log_dir: "{{ data_dir }}/log/pods"
####目前暂时不支持 containerd 的数据目录不加{{data_dir}}
#### 建议数据盘挂载到/opt 下，kubelet 和 containerd 都使用同一个分区，不然 ephemeral-storage limit 功能无法使用
####数据目录####
```

inventory

inventory 为 kubernetes 集群节点的 ssh 登录配置。

参数	说明	示例
ansible_port	主机 ssh 登录端口号	22
ansible_user	主机 ssh 登录用户名	
ansible_ssh_pass	主机 ssh 登录密码	
ansible_sudo_pass	sudo 密码	
ansible_ssh_private_key_file	如果使用 private key 登录	必须为/kubespray/config/目录，文件名可以变化，这个目录为运行 ansible 容器的目录,并非宿主机的目录，对应的宿主机的目录为 kubeplay/config/kubespray/，需要把 id_rsa 拷贝到 kubeplay/config/kubespray/下
ansible_host	节点 IP	
kubernetes_cluster_used_ip	该节点的 kubernetes 服务使用的 IP	

- yaml 格式

```
#### Cluster nodes inventory info

#####注意缩进不要使用 tab，所有缩进必须全部使用空格
inventory:
  all:
    vars:
      #####默认的 ssh 相关变量，也可以为某个节点单独配置
      ##ansible_ssh_pass: ssh 密码 就是用该用户(ansible_user)ssh 登录的密码
      ##ansible_sudo_pass: sudo 密码 就是用 ansible_user 执行 sudo 命令的时候，输入的密码
      ##一般来说 这两密码是相同的
      ansible_port: 22
      ansible_user: root
      ansible_ssh_pass: Password
      ## ansible_sudo_pass: 123
      # ansible_ssh_private_key_file: /kubespray/config/id_rsa
```



```

hosts:
  node1:
    ### 注意 一定要是系统内的IP 地址，不可以写域名和VPCIP、NAT IP 等，
    如果确实有需要配置VPC IP 或者NAT IP，就要配置kubernetes_cluster_used_ip 为
    系统内的IP
    ansible_host: 172.20.0.21
    ## kubernetes 集群的各组件/etcd/ 网络插件使用的IP 地址，默认为ansib
    le_host，如果需要自定义单个节点的IP，就打开取消下面的注释，定义IP 地址
    ### 注意 一定要是系统内的IP 地址，不可以写域名和VPC IP、NAT IP 等
    ## kubernetes_cluster_used_ip: 192.168.1.1
  node2:
    ### 注意 一定要是IP 地址，不可以写域名
    ansible_host: 172.20.0.22
    ## 单个节点配置变量，此变量仅作用于单个节点，优先级高于上面的vars 变
    量

    ## ansible_port: 22
    ## ansible_user: root
    ## ansible_ssh_pass: Password
    ## ansible_sudo_pass: 123
    ## ansible_ssh_private_key_file: /kubespray/config/id_rsa
  node3:
    ### 注意 一定要是IP 地址，不可以写域名
    ansible_host: 172.20.0.23
  node4:
    ### 注意 一定要是IP 地址，不可以写域名
    ansible_host: 172.20.0.24
  children:
    kube_control_plane: ###master 节点 建议为两个或者三个 只要集群规模大
    于等于3，就配置三个master 节点，只需要写上hosts 下的name 加冒号，不需要在后
    面加IP，（ansible 语法）
    hosts:
      node1:
      node2:
      node3:
    kube_node:
    hosts:
      node1:
      node2:
      node3:
      node4:
    etcd: ###etcd 节点 建议为1 个或者三个 只要集群规模大于等于3，就配置三
    个etcd 节点
    ###etcd 节点的数据不能等于两个
    hosts:
      node1:
      node2:
      node3:
    kubernetes_cluster:

```



```

    children:
      kube_control_plane:
      kube_node:
    gpu:
      hosts: {}
    calico_rr: {}

```

###calico RR 节点，在集群规模大于50 时启用，因为calico 默认为nodeToNodeMesh 模式，也就是节点全互联模式，以50 个节点为例，全互联模式每个节点都会和其他49 个节点都建立BGP 连接，也就是会有 $50 \times 49 = 2450$ 条BGP 连接，一般建议在50 个节点以上时就配置RR 节点，找几个配置较高的k8s 节点配置成RR 节点，定义RR 节点后会关闭节点全互联模式，而是所有的非RR 节点都和RR 节点建立BGP 连接，互相宣告BGP 路由表，RR 节点之间也会建立BGP 连接，这样BGP 连接数量就会大幅减少。

```

# hosts:
#   node1:
# vars:
#   cluster_id: "1.0.0.1"

```

default

以下几个默认的参数在没有特殊要求的情况下不建议修改，直接保持默认即可。
ntp_server 参数为默认值时会自动替换成 compose 中的 internal_ip 值；
registry_ip 和 offline_resources_url 这两个参数会根据 compose 中的参数自动生成无需修改。

参数	说明	示例
ntp_server	ntp 时钟同步服务器域名或 IP	-
registry_ip	镜像仓库节点 IP	-
offline_resources_url	提供离线资源下载的 URL 地址	-
offline_resources_enabled	是否为离线部署	true
generate_domain_cert	是否为镜像仓库域名生成自签证书	true
image_repository	镜像仓库的 repo 或 project	library
download_container	是否在所有节点 pull 下所有组件的镜像	false

default:

```

# NTP server ip address or domain, default is internal_ip
ntp_server:
  - internal_ip ##默认即可， 一般无需修改
# Registry ip address, default is internal_ip
registry_ip: internal_ip ##默认即可， 一般无需修改
# Offline resource url for download files, default is internal_ip:nginx_http_port
offline_resources_url: internal_ip:nginx_http_port ##默认即可， 一般无需修改

```

```

registry_domain: imagerepo_domain:registry_https_port ##默认即可， 一

```

般无需修改

```
# Use nginx and registry provide all offline resources
offline_resources_enabled: true ##默认即可， 一般无需修改
# Image repo in registry
image_repository: library ##默认即可， 一般无需修改
# Kubespray container image for deploy user cluster or scale
kubespray_image: "kubespray" ##默认即可， 一般无需修改
# Auto generate self-signed certificate for registry domain
generate_domain_cert: true ##默认即可， 一般无需修改
# Set false to disable download all container images on all nodes
download_container: false ##默认即可， 一般无需修改

# enable support hubble in cilium
cilium_enable_hubble: false ##默认即可， 一般无需修改
# install hubble-relay, hubble-ui
cilium_hubble_install: false ##默认即可， 一般无需修改
# install hubble-certgen and generate certificates
cilium_hubble_tls_generate: false ##默认即可， 一般无需修改
# Kube Proxy Replacement mode (strict/probe/partial)
cilium_kube_proxy_replacement: probe ##默认即可， 一般无需修改
```

基本操作

安装

```
sudo bash install.sh
```

调试 shell 脚本并安装:

```
sudo bash -x install.sh ##调试 shell 脚本, 可以输出 shell 脚本的所有日志
```

安装过程中可能出现多次日志暂停, 是因为脚本中有重启 containerd 等操作, 这时只需要 ctrl+c 然后重新 log

```
sudo nerdctl logs -f kubespray-runner
```

如果安装过程中出现退出的情况, 可以修复问题后, 执行

```
sudo nerdctl start kubespray-runner
sudo nerdctl logs -f kubespray-runner
```

扩容节点

只能扩容 node, 不能扩容 etcd 节点和 master 节点。

添加节点:

```
cd /opt/kubeplay
vi config/kubespray/inventory
```

在 hosts 和 kube_node 下添加节点信息:

```
sudo bash install.sh add-node <NODE_NAME>,<NODE_NAME>
```

示例:

```
sudo bash install.sh add-node node2,node3
```

删除节点:

```
sudo bash install.sh remove-node <NODE_NAME>
```

示例:

```
sudo bash install.sh remove-node node2
```

避免误操作导致节点被移除，需要手动删除一段脚本才可以正常运行.

组件版本

addon	version	用途
kubernetes	v1.21.5	kubernetes
containerd	v1.4.9	容器运行时
etcd	v3.4.13	etcd 服务
crictl	v1.21.0	CRI CLI 工具
pause	3.3	pause 容器镜像
cni-plugins	v0.9.1	CNI 插件
calico	v3.18.4	calico
autoscaler	1.8.3	DNS 自动扩缩容
coredns	v1.8.0	集群 DNS 服务
flannel	v0.14.0	flannel
nginx	1.19	node 节点反向代理 APIserver
canel	calico/flannel	集成 calico 和 flannel
helm	v3.6.3	helm CLI 工具
nerdctl	0.14.0	containerd CLI 工具
registry	v2.7.1	提供镜像下载服务
skopeo	v1.4.0	镜像搬运工具

kuebplay 安装 kubernetes 支持的操作系统:

注意：并非安装 deepflow 支持的操作系统

distribution	version	arch
CentOS	7/8	amd64/arm64
Redhat	7	amd64/arm64

Debian	10/9	amd64/arm64
Ubuntu	20.04/18.04	amd64/arm64
Fedora	33/34	amd64/arm64

以下操作暂时不可用也没有使用场景，先放到最下面

移除集群

```
sudo bash install.sh remove-cluster
```

移除所有组件

```
sudo bash install.sh remove
```

附：

sudo 配置：

如果用普通用户 sudo 安装，需要有 sudo ALL 权限

```
vi /etc/sudoers
```

修改 sudo PATH 变量,添加"/usr/local/bin

```
Defaults    secure_path = /sbin:/bin:/usr/sbin:/usr/bin:/usr/local/bin
```

添加到“# %wheel ALL=(ALL) NOPASSWD: ALL”后面

```
<USER>      ALL=(ALL)      NOPASSWD:ALL
```

快速调试/使用 [ansible](#)

```
nerdctl run -it --rm -v /opt/kubeplay/kubespray:/kubespray -v /opt/kubeplay/config/kubespray:/kubespray/config -v /opt/kubeplay/kubespray/ansible.cfg:/etc/ansible/ansible.cfg $(nerdctl images | awk '{print $1":"$2}' | grep '^kubespray:*' | sort -r --version-sort | head -n1) bash
ansible -f 10 --become --become-user=root -i config/inventory -e @config/env.yml all -m ping
```

恢复 etcd 集群数据

确保/var/backups/下有 etcd-snapshot.db 文件，且该文件大小不为 0，可以用 du -sh 命令测试

以 kubernetes 数据目录为/opt 为例，则 etcd 的数据目录为/opt/lib/etcd,

在所有 etcd 节点操作

```
systemctl stop etcd
```

```
mv /opt/lib/etcd /opt/lib/etcd_bak
```

为了保证数据一致，需要将一个 etcd 节点的 etcd-snapshot.db 文件 scp 到其他 etcd 节点,文件路径一般在 /var/backups/etcd-snapshot.db

```
scp /var/backups/etcd-snapshot.db xxx:/var/backups/etcd-snapshot.db
```

scp 完成后在所有 etcd 节点执行

```
bash /etc/etcd-restore.sh
```

```
systemctl restart etcd ###此命令在所有节点上的 etcd 都启动完成后结束
```

检查 etcd 日志和 kubernetes 数据

```
journalctl -xeu etcd
```

containerd 基本操作

containerd 有 namespace 的概念, 常见的有 kubernetes.io 下为 kubernetes 的容器和镜像, moby 下为 docker 的容器和镜像, default 下为默认的资源,

nerdctl 默认操作 default 下的资源

nerdctl 操作和 docker 大部分比较像

```
nerdctl -n k8s.io ps
```

```
nerdctl -n k8s.io images
```

拉取非安全仓库镜像

```
nerdctl --insecure-registry=true pull xxxxx
```

目前 nerdctl 暂时不支持推送镜像到非安全仓库 其他推送办法

例如 skopeo

拷贝镜像

```
skopeo copy --src-tls-verify=false --dest-tls-verify=false docker-daemon:registry:latest docker://127.0.0.1:5000/library/registry:cccc
```

拷贝镜像到阿里云仓库

```
skopeo copy docker-daemon:registry:latest docker://registry.cn-hangzhou.aliyuncs.com/kubespray-image/registry:latest
```

从 registry 拷贝到阿里云仓库

```
skopeo copy docker://127.0.0.1:5000/xxxx/xxxx:xxxx docker://registry.cn-hangzhou.aliyuncs.com/xxxxx/xxxxx:xxx --dest-creds=USERNAME:PASSWORD --src-tls-verify=false
```

tar 包到非安全仓库

```
skopeo copy --insecure-policy --dest-tls-verify=false docker-archive:/root/kubespray-v0.1.0-rc.18.tar docker://127.0.0.1:5000/kubespray:v0.1.0-rc.18
```

虚拟磁盘使用方法

一般来说 containerd 的数据盘不建议太小, 有条件的项目建议在 100G 左右。

dd 一个 100G RAW 格式的虚拟磁盘文件作为虚拟磁盘:

```
dd if=/dev/zero of=/opt/containerd_vdisk.img bs=1G count=0 seek=100
```

格式化设备为文件系统:

```
mkfs.xfs -f -n ftype=1 /opt/containerd_vdisk.img
```

挂载到文件系统中:

```
echo "/opt/contaienrd_vdisk.img /opt/lib/contaienrd xfs defaults,loop
0 0" >> /etc/fstab
mount -a
xfs_repair -L /opt/contaienrd_vdisk.img
```

修改 containerd 配置, 将 containerd 的 root 目录修改为虚拟磁盘挂载的目录:

```
vi /etc/containerd/config.toml

root = "/opt/lib/contaienrd"
```

多次 reboot 查看系统是否能正常启动, containerd 是否能正常启动、kubernetes 集群是否正常。如果无法启动, 可以执行如下命令, 然后重新启动:

```
xfs_repair -L /opt/contaienrd_vdisk.img
```

虚拟磁盘已知问题:

- 两层文件系统格式化, 性能可能较差。
- 可能会因为 xfs 文件系统有脏数据导致系统无法成功启动的情况。

解决办法:

```
xfs_repair -L /opt/contaienrd_vdisk.img
```

kubeplay 工作目录误删除恢复方法

kubeplay 的工作目录保存着这个集群的配置、镜像等, 后续的集群更新、扩容等需要该目录, 并且该目录为 kubernetes 集群的镜像仓库 registry 工作目录, 删除后可能无法正常拉取镜像。

恢复方法:

由于丢失了 rootCA 的 key, 所以需要重新签发 CA 证书。

需要先解压 kubeplay 压缩包, 恢复 config.yaml 文件, 按照之前的配置重新写上。

```
cd /opt/kubeplay
cp config-sample.yaml config.yaml
vi config.yaml
##### 手动恢复配置
```

在 kubeplay 部署工具节点执行命令:

```
cd /opt/kubeplay/
KUBE_ROOT=$(cd "$(dirname "$0")" && pwd)
KUBESPRAYDIR="${KUBE_ROOT}/kubespray"
CERTS_DIR="${KUBE_ROOT}/config/certs"
```

```

CONFIG_FILE="${KUBE_ROOT}/config.yaml"
RESOURCES_NGINX_DIR="${KUBE_ROOT}/resources/nginx"
IMAGEREPO_DOMAIN=$(yq eval '.compose.imagerepo_domain' ${CONFIG_FILE})
DOMAIN=$(echo ${IMAGEREPO_DOMAIN} | sed 's/[^.]*.*./')
sudo mkdir -p ${CERTS_DIR} ${RESOURCES_NGINX_DIR}/certs
CAROOT=${CERTS_DIR} sudo mkcert -install
CAROOT=${CERTS_DIR} sudo mkcert -key-file ${CERTS_DIR}/domain.key -cert
-file ${CERTS_DIR}/domain.crt ${IMAGEREPO_DOMAIN} ${DOMAIN}
sudo /bin/cp -f ${CERTS_DIR}/rootCA.pem ${RESOURCES_NGINX_DIR}/certs/rootCA.crt
sudo nerdctl compose -f compose.yaml down
sudo nerdctl compose -f compose.yaml up -d

```

在所有 Kubernetes 节点执行：

由于重新签发了 CA 证书，需要所有节点的系统信任该证书并重启 containerd，让 containerd 信任该证书。

拷贝/opt/kubeplay/config/certs/rootCA.pem 文件到所有 Kubernetes 节点的指定位置：

- Debian 家族系统： /usr/local/share/ca-certificates/kubeplay-ca.crt
- RedHat 家族系统： /etc/pki/ca-trust/source/anchors/kubeplay-ca.crt

执行命令：

- Debian 家族系统


```

update-ca-certificates
systemctl restart containerd

```
- RedHat 家族系统


```

update-ca-trust extract
update-ca-trust
systemctl restart containerd

```

检查是否可以正常拉取镜像。

基础部署

部署框架介绍

- 使用 docker image 发布软件包
- 使用 Kubernetes 编排进程运行状态
- 使用 Helm 完成部署

规格

- 由于运行 Kuernetes 本身会占用一部分资源，要求最小配置为 4C16G。

服务器角色介绍

- 主要角色
 - **master_controller**: 用于标记主区域控制器节点, 至少有一个, 与 **slave_controller** label 互斥
 - **slave_controller**: 用于标记从区域控制器节点, 仅从区域部署时使用 **slave_controller**, 至少有一个, 与 **master_controller** label 互斥
 - **tsdb**: 用于部署 ClickHouse 数据节点, 至少有一个, NPB 模式下置为运维模式, 可以与 **controller** 位于同一台服务器上, 用于存储时序数据, 考虑到性能推荐使用 SSD
 - **dfdb**: 用于指定运行 MySQL 和 InfluxDB 的控制器节点。
 - 有共享存储: 可标记多个 **controller** 上部署 MySQL。
 - 无共享存储: 仅支持标记一个固定节点, 需要与 **master_controller** 或 **slave_controller** (从区域仅部署 InfluxDB) 位于同一台服务器上。
- 其他角色
 - **elasticsearch-warm**: 服务器会安装 elasticsearch
 - **trident**: 【可选】用于部署 ESXi 环境中的采集器虚拟机
 - **dedicated-trident**: 【可选】用于部署专用物理机采集器

依赖

警告: 确保所有控制器、数据节点、专属服务器采集器、ESXi 采集器的 **disable_ipv6** 参数都为 0: `sysctl -a | grep "disable_ipv6"`。开启 IPV6 参考文档底部 FAQ。

- 部署环境需要确保 Kubernetes 正常工作, 参考《准备工作-安装部署 kubernetes》
- 每个节点需部署 ntp 服务, 并提供 123 端口访问, 部署 ntp 可参考底部 faq
- 当前使用固定路径提供存储目录, 需手动将磁盘挂载至以下目录, 若未挂载, 将自动创建, 此时无法使用数据盘
 - **pcap**: /mnt/pcap
 - **elasticsearch**: /mnt/es
 - **influxdb-data**: /mnt/influxdb/data
 - **influxdb-meta**: /mnt/influxdb/meta
 - **clickhouse-data**: /mnt/clickhouse
 - **mysql**: /var/lib/mysql

内核更新到最新版本

使用 CentOS 7, 内核更新到最新的 4.19。

- 获取内核安装包方式

- 从 ISO 包中的 `extra-artifacts/kernel-installation-package` 文件夹下获取。
- 安装:

```
# 如果已经安装了4.19 内核，先把旧版本删除，然后安装新版本。
/usr/bin/rpm -e kernel-4.19.17-1.el7.x86_64 --nodeps
/usr/bin/rpm -ivh --force ./kernel-4.19.17-1.el7.x86_64.rpm --nodeps
```

- 验证：重启机器待完成后，使用 `uname` 检查是否已使用新的 Linux Kernel。

```
uname -r
# 4.19.17-1.el7.x86_64
```

部署 DeepFlow

DeepFlow Deploy 是基于 Kubernetes 和 Helm 的部署工具包。

域名规划

多区域部署时需要，一个主区域时可跳过该部分内容

注意：域名规划中写 `/etc/hosts` 操作于 2022/01/20 已废弃，采用内建负载均衡器方式提供主从区域互访。参考《从区域部署》。

标记节点 label

- 标记节点

```
kubectl label node [node name] [role]=enable
```

- 清除标记

```
kubectl label node [node name] [role]-
```

- Label 参考示例：

```
# label 含义参考上文【服务器角色介绍】
# 仅一台机，AllInOne 必备 label 示例：
master_controller
tsdb
dfdb
elasticsearch-warm
dedicated-trident [可选]
trident [可选]
# 非 AllInOne ，与上述区别：
master_controller      可以打到多个节点上
dfdb                   必须选一台 master_controller，增加 dfdb label
# 从区域与上述区别：
```

只用 `slave_controller`，不需要 `master_controller`
其他 `label` 与主区域无异

离线部署模式

基础准备

注意：重复部署时，如 `/usr/local/deepflow` 文件夹已存在，需先使用 `mv` 备份，防止影响本次部署 `mv /usr/local/deepflow/ /usr/local/deepflow_$(date +%Y%m%d)_bak`

挂载 ISO

```
mount -o ro deepflow-docker-release-*.iso /media
```

创建子目录及父目录

```
mkdir -p /usr/local/deepflow/debug
```

拷贝 ISO 包所有文件至指定目录，包含 `trident` 安装包

```
/bin/cp -rf /media/* /usr/local/deepflow/
```

如升级版本时需从 ISO 中合并仓库，则合并后执行命令重启仓库 `pod`

```
cd /opt/kubeploy/ && nerdctl compose -f compose.yaml down && nerdctl compose -f compose.yaml up -d
```

创建离线仓库

- `kubeploy` 安装的 Kubernetes 集群 该方式部署的 K8s 集群默认已配置 DeepFlow 的 registry 为信任的镜像仓库，无须修改 `containerd` 的配置文件!!!

```
vi /opt/kubeploy/compose.yaml
```

将下面的注释取消即可

```
deepflow-registry:
  image: registry:2.7.1
  container_name: deepflow-registry
  restart: always
  network_mode: host
  volumes:
    - /usr/local/deepflow/registry:/var/lib/registry
    - ./config/deepflow-registry.yaml:/etc/docker/registry/config.yaml
    - ./config/certs/domain.crt:/etc/docker/registry/domain.crt
    - ./config/certs/domain.key:/etc/docker/registry/domain.key
```

DeepFlow 的镜像 tag 为安装 Kubernetes 集群的配置文件 `config.yaml` 的 `compose.imagerepo_domain + registryr` 端口号。

例如下面的域名为 `kube.registry.local`，registry 端口为 `5000`，那么 DeepFlow 的 repository 为 `kube.registry.local:5000`

无需其他任何配置

```
compose:
  imagerepo_domain: kube.registry.local
```

然后重启 compose 文件中的三个容器

```
cd /opt/kubeplay/
sudo nerdctl compose -f compose.yaml down
sudo nerdctl compose -f compose.yaml up -d
```

- 其他 Kubernetes 集群

```
docker load -i /usr/local/deepflow/registry.tar
docker run -d -p 5000:5000 -v /usr/local/deepflow/registry:/var/lib
/registry --restart=always --name registry registry:latest
```

离线非信任仓库配置方法

每个 Kubernetes 的 node 节点都需要修改配置，根据运行时不同，分为 Docker 和 Containerd，使用 kubeplay 安装的 K8s 集群无需配置。

- Docker 配置：/etc/docker/daemon.json

```
# 需修改 http://[registry_ip]:5000 为上步创建的仓库地址
cat << EOF > /etc/docker/daemon.json
{
  "insecure-registries":["http://[registry_ip]:5000"]
}
EOF
# 修改后重启 Docker 进程: systemctl daemon-reload && systemctl restart docker
```

- Containerd 配置：/etc/containerd/config.toml

```
# 需修改 http://10.10.10.10:5000 为上步创建的仓库地址
# 在[plugins."io.containerd.grpc.v1.cri".registry.mirrors]下添加，注意修正缩进:
    [plugins."io.containerd.grpc.v1.cri".registry.mirrors."10.10.10.10:5000"]
        endpoint = ["http://10.10.10.10:5000"]
    [plugins."io.containerd.grpc.v1.cri".registry.configs."10.10.10.10:5000".tls]
        insecure_skip_verify = true
# 修改后重启 Containerd 进程: systemctl daemon-reload && systemctl restart containerd
```

修改 DeepFlow 部署参数

- 按规划设计及注释修改为当前环境信息:
/usr/local/deepflow/templates/values-custom.yaml

注意：不要直接修改 values.yaml，所有自定义参数需按相同格式放置在 values-custom.yaml 中

部署

部署命令

命令说明：- ALL 代表 DeepFlow 自研组件 + 其他组件（mysql redis grafana rabbitmq 等）- DeepFlow 代表 DeepFlow 自研组件，不卸载 mysql redis grafana rabbitmq 等中间件 - 卸载均不删除数据，数据保留在宿主指定目录下，如有需要自行删除

```
[root@K8s /]# /usr/local/deepflow/bin/deepflow-deploy -h
-i, --install          Install all          [安装 DeepF
low 所有依赖，但默认不初始化数据库]
-m, --middleware       Install middleware
-j, --job              Run Jobs to init database [初始化数据
库 注意：会清空原数据库，恢复初始状态，请谨慎操作，并提前备份数据库]
-d, --deepflow         Install DeepFlow
-u, --upgrade          Upgrade all          [更新所有应
用 yaml 改动]
-uo, --upgrade-one     Upgrade One          [更新单个应
用 yaml 改动]
-ud, --upgrade-deepflow Upgrade DeepFlow    [更新 DeepF
low 业务组件 yaml 改动]
-e, --erase            Uninstall all
-em, --erase-middle    Uninstall middleware
-ej, --erase-job       Uninstall job
-ed, --erase-deepflow  Uninstall DeepFlow
```

一键部署

先执行

```
/usr/local/deepflow/bin/deepflow-deploy -i
```

仅首次安装，或需要重置数据库时执行

后执行

```
/usr/local/deepflow/bin/deepflow-deploy -j
```

一键卸载

```
/usr/local/deepflow/bin/deepflow-deploy -e
```

卸载默认不删除任何数据，如需删除数据请参考【依赖】中的数据路径，手动删除

一键更新

更新所有应用

```
/usr/local/deepflow/bin/deepflow-deploy -u
```

更新单个应用，应用名称为 `/usr/local/deepflow/templates/` 中目录名称
`/usr/local/deepflow/bin/deepflow-deploy -uo cloud-agent`

Python Debug

- 开启 debug

```
/usr/local/deepflow/bin/deepflow-debug -d [仓库地址] [项目名称] [版本号]
example:
/usr/local/deepflow/bin/deepflow-debug -d docker-hosted.nexus.yunshan.net alarm 5.7.0
```

- 关闭 debug

```
/usr/local/deepflow/bin/deepflow-debug -r [仓库地址] [项目名称] [版本号]
example:
/usr/local/deepflow/bin/deepflow-debug -r docker-hosted.nexus.yunshan.net alarm 5.7.0
```

Patch 升级工具

功能

- 支持容器镜像及 `post_patch.sh`、`mysql_issu.sql` 的升级
- 支持专属采集器、ESXI 采集器自动升级
- 不支持 trident kvm 采集器自动升级

升级流程

- 获取 Patch 压缩包
- 执行 patch 命令

```
/usr/local/deepflow/bin/deepflow-patch [Patch 包路径]
example:
/usr/local/deepflow/bin/deepflow-patch /root/15-20211213-16728-BUG.tar.gz
```

mntnct

5.7.3 版本之后，mt 工具随 DeepFlow 组件一起部署为服务，mt 命令需进入到容器内执行。

- 进入 mt 容器：

```
`MT_POD=$(kubectl get pods -n deepflow | grep mntnct | awk '{print $1}')
```

```
`kubectl exec -it $MT_POD -n deepflow sh`
```

- 测试命令

mt help

DeepFlow 计费模式 [可选]

DeepFlow 计费模式默认为授权模式，若需要修改计费模式，需要如下操作：- 切换前需要在页面上将所有授权文件删除 - 修改部署参数
/usr/local/deepflow/templates/values-custom.yaml 中对应的选项 ... rear-end # 计费方式: license/voucher # license: 授权模式 # voucher: 按量计费模式 billing_method = voucher ... - 修改完成后重新生成 configmap: /usr/local/deepflow/bin/deepflow-deploy -u

- 按照先后次序依次 kubectl delete -n deepflow xxxx 主控制器上的 df-web、warrant、manager 容器
- 如果已经存在控制器或数据节点，切换后需要手工取消控制器、数据节点的运维模式
- 如果已经存在采集器，需要手动注册采集器、手动启用采集器

注意：不支持先使用按量计费，再使用授权模式这种计费方式切换。

DeepFlow 按量计费邮件配置 [可选]

若需要发送计费邮件，需修改 manager 配置文件，修改 /usr/local/deepflow/templates/manager/templates/configmap.yaml 中对应的选项：

```
email:
  # 邮件服务器地址
  smtp_host: smtp.exmail.qq.com
  # 邮件服务器端口
  smtp_port: 25
  # 邮件接收地址
  to_emails:
    - name1@yunshan.net
    - name2@yunshan.net
  # 邮件服务器用户名
  user: server@yunshan.net.cn
  # 邮件服务器用户密码
  password: xxxxxxxxxx
  # 邮件发送方式(plain/ssl/tls)
  type: plain
```

DeepFlow 告警邮件配置 [可选]

若需要发送告警邮件，参考 values.yaml 中 postman 部分，将配置添加进 values-custom.yaml 中，再使用 deepflow-deploy -uo postman 更新组件配置。

```
postman:
  emailConfig:
```

```
## 是否开启 ntlm 认证， 默认值 False，不开启；True 为开启
ntlm_enabled: False
## NTLM 认证协议的登录用户，格式为 domain\username，开启 NTLM 认证后配置。
## 默认 user 配置选项的截取内容，默认值不支持 user 邮箱的域名存在子域名，
需配置该选项
ntlm_login_name:
## NTLM 认证协议的登录用户密码， 默认和 password 配置一致
ntlm_password:
## SMTP 超时时间
smtp_timeout: 300
env_lang: ENGLISH
## security 有效值 ssl/tls/none，默认是 none
security: none
host : smtp.exmail.qq.com
port : 25
user : stats01@yunshan.net.cn
password : FIXME-password
```

DeepFlow 采集器 C 类授权 [可选]

自动发现时采集器默认会使用 A 类或 B 类授权，但管理员可将其手动调整为使用 C 类授权。一个宿主机上的所有采集器仅需要消耗一个 C 类授权，因此一旦有一个采集器被设置为使用 C 类授权后，这个宿主机上被发现的其他采集器都会自动的使用 C 类授权。

仅当采集器运行环境能关联至宿主机时（例如容器采集器可通过容器节点对应的云服务器关联至宿主机、Workload 采集器可通过云服务器关联至宿主机），一个采集器才能使用 C 类授权。

采集器可以使用的授权类型如下：

采集器类型	授权类型
KVM	A 类、C 类
ESXI	A 类、C 类
Hyper-V	A 类、C 类
专属服务器	A 类
隧道解封装	A 类
容器-V	A 类、C 类
容器-P	A 类
Workload-V	A 类、B 类、C 类
Workload-P	B 类

若需要修改默认授权类型，需修改 manager 配置文件，修改 /usr/local/deepflow/templates/manager/templates/configmap.yaml 中对应的选项：

指定授权类型仅对支持该类型的采集器生效
可选值：空/A/B/C，默认为空
空：消耗采集器默认的 A 类或 B 类授权
A：支持 A 类授权的采集器，自动发现时会消耗 A 类授权
B：支持 B 类授权的采集器，自动发现时会消耗 B 类授权
C：支持 C 类授权的采集器，自动发现时会消耗 C 类授权
original_authorization: C

初始化 ES 索引模版

操作步骤

此步骤自 6.0.0 版本后已更新为自动配置，无需手动操作，以下流程仅供手动排查问题参考

手工配置

上传/更新索引模版：

```
curl -H "Content-Type: application/json" -XPUT "http://`kubectl get svc -n deepflow elasticsearch -o jsonpath='{.spec.clusterIP}'`:20042/_template/deepflow_system_log" -T /opt/deepflow_system_log_es_template.json
```

deepflow_system_log_es_template.json 文件内容：

```
{
  "order": 0,
  "index_patterns": [
    "deepflow_system_log__0_*"
  ],
  "settings": {
    "index": {
      "routing": {
        "allocation": {
          "require": {
            "box_type": "warm"
          }
        }
      }
    },
    "mapper": {
      "dynamic": "false"
    },
    "refresh_interval": "10s",
    "number_of_shards": "11",
    "translog": {
```



```

        "durability": "async"
    },
    "number_of_replicas": "0"
}
},
"mappings": {
    "events": {
        "_source": {
            "enabled": true
        },
        "properties": {
            "severity": {
                "store": true,
                "type": "keyword"
            },
            "syslogtag": {
                "store": true,
                "type": "keyword"
            },
            "module": {
                "store": true,
                "type": "keyword"
            },
            "host": {
                "store": true,
                "type": "keyword"
            },
            "message": {
                "store": true,
                "type": "keyword"
            },
            "type": {
                "store": true,
                "type": "keyword"
            },
            "userid": {
                "store": true,
                "type": "keyword"
            },
            "timestamp": {
                "format": "epoch_second",
                "store": true,
                "type": "date"
            }
        }
    }
},
"aliases": {}
}

```

FAQ

Trident 安装包在哪里？

ISO 包中的 `extra-artifacts` 文件夹下

如何从在线仓库 `pull` 镜像，再推送至本地仓库

适用于 QA 从在线环境获取最新镜像

拉取

```
docker pull docker.nexus.yunshan.net/apientry:5.7.3
```

重命名

```
docker tag docker.nexus.yunshan.net/apientry:5.7.3 127.0.0.1:5000/apientry:5.7.3
```

推送

```
docker push 127.0.0.1:5000/apientry:5.7.3
```

如何启动 ntp 服务

检查各 Kubernetes 节点的 UDP123 端口是否监听

如果 UDP 123 端口是 deepflow namespace 下的 chrony pod 监听的，需要关闭/删除 chrony 的 daemonset

```
ss -anulp|grep 123
```

检查是否已存在 chrony 服务

```
systemctl status chronyd
```

```
ss -anulp | grep `systemctl status chronyd | grep PID | awk '{print $3}`
```

如未安装则执行

```
yum install -y chrony
```

```
systemctl enable chronyd --now
```

添加 `allow all`（有此配置时 chronyd 将提供 ntp 123 端口服务）

```
echo "allow all" >> /etc/chrony.conf
```

重启 chronyd 进程

```
systemctl restart chronyd
```

再次检查

```
ss -anulp | grep `systemctl status chronyd | grep PID | awk '{print $3}`
```

如何开启 IPV6

Centos

```
echo "net.ipv6.conf.all.disable_ipv6" >> /etc/sysctl.d/98-kernel.conf
```

```
echo "net.ipv6.conf.all.disable_ipv6=0" >> /etc/sysctl.d/98-kernel.conf
```

```
echo "net.ipv6.conf.default.disable_ipv6=0" >> /etc/sysctl.d/98-kernel.conf
```

```
echo "net.ipv6.conf.lo.disable_ipv6=0" >> /etc/sysctl.d/98-kernel.conf
```

```
sysctl -p /etc/sysctl.d/98-kernel.conf
```

如何检查 kubelet 的驱逐配置？

当使用公有云集群或者其他方式自建集群时，需检查磁盘空间驱逐条件，是否满足 DeepFlow 需求。

查询 kubelet 配置方法：

```
systemctl status kubelet -l
```

- 查看启动命令中是否有 `--config` 参数，如果有该参数，就查看该文件中是否有如下配置项：

```
evictionHard:
  memory.available:
  nodefs.available:
  nodefs.inodesFree:
  imagefs.available:
```

- 查看启动命令中是否有 `--eviction-hard=imagefs.available<xx,memory.available<xx,nodefs.available<xx,nodefs.inodesFree<xx` 参数 如果 `nodefs.available`，`imagefs.available` 两个配置项的百分比值高于百分之 10 或者数量值除以磁盘大小后高于百分之 10，需要修改为小于百分之 10。

例如 200G 磁盘，配置值为“40Gi”或者“15%”，则认为是大于百分之 10，需要调整为小于百分之 10，建议值为“5%”或者“10Gi”。

配置示例：配置文件配置方法：

```
evictionHard:
  memory.available: "512Mi"
  nodefs.available: "10Gi"
  nodefs.inodesFree: "5%"
  imagefs.available: "10Gi"
```

启动参数配置方法： `--eviction-`

```
hard=imagefs.available<10Gi,memory.available<512Mi,nodefs.available<10Gi,nodefs.inodesFree<5%
```

CHANGELOG

部署工具包更新记录

Added	— 这里记录新增加了哪些功能 / 接口
Changed	— 功能变更
Deprecated	— 不建议使用的功能 / 接口，将来会删掉
Removed	— 之前不建议使用的功能 / 接口，这次真的删掉了
Fixed	— 这里记录解决了哪些问题
Others	— 其他改进

6.0.0

[Added]

- `-uo` 命令支持升级单个组件，无需等待所有组件更新 `/usr/local/deepflow/bin/deepflow-deploy -uo log-cleaner`
- `-u/-uo` 命令增加 `--install` 属性，如未安装则安装，如已存在则升级，便于操作单个组件安装或升级
- MySQL 支持在 `values.yaml` 中填写外部 IP 类型 MySQL
- MySQL 支持在 `values.yaml` 中挂载 PVC
- InfluxDB 支持在 `values.yaml` 中挂载 PVC
- 支持配置 https 证书，详情见 `values.yaml` 注释及《页面定制配置》文档
- 邮箱、oidc 配置支持在 `values.yaml` 中修改
- MySQL 每日数据备份，默认在 `/usr/local/deepflow/mysql` 目录下
- 修改 `configmap` 后自动重建应用，无需手动 `delete pod`（单节点时可能由于滚动策略不同，出现 Pending 情况，手动删除旧的即可）
- 自动创建 es 索引模板，无需手动配置
- 自动 Patch 工具，参考 Patch 升级文档
- 主动区域互访方式改变，无需配置 `/etc/hosts` 自定义域名，使用内建的四层负载均衡器提供专用 `service + endpoint`，并对目标 IP 做健康检查

[Changed]

- 部署参数模板调整为 2 个 `values.yaml`、`values-custom.yaml`，不要直接修改 `values.yaml`，所有自定义参数需按相同格式放置在 `values-custom.yaml` 中
- 卸载时默认不删除 `deepflow namespaces`，防止命名空间下有其他资源，比如 `pv pvc secret` 等被删除
- `-j` 重置数据库命令增加确认操作，防止误操作清空数据库
- `front-end` 修改为 `daemonset`，部署后使用任意 `master_controller ip` 即可访问【优先使用第一个节点】
- 原 `apientry` 组件拆分成 `apientry df-web-core df-help` 三个组件独立部署

[Removed]

- 取消 `chrony` 时间同步工具的部署，由 K8s 安装脚本在宿主节点直接提供。

5.7.3

高可用初始化

高可用（HA）支持

前提条件

- k8s 集群能支持使用动态 PVC 创建共享存储卷

存储 HA

如：MySQL Influxdb ES 数据

- 仅在共享存储的情况下支持 HA（例如：NAS 共享文件存储、OSS 共享对象存储）

- 推荐使用性能较高的 NAS 存储，如 NAS 存储有性能型，则选择性能更好的类型。
- 不推荐自建 NFS 共享存储，需客户自行保证自建 NFS 的可用性。

存储方案选型

- NAS 共享存储 【首选】
 - 优势：通常提供有高性能 NAS，且支持挂载至多个 POD，如遇节点宕机，可快速在其他节点启动 MySQL 并使用相同数据
 - 风险：暂无
- OSS 对象存储
 - 优势：同上
 - 风险：性能较低，不推荐
- 云盘/RBD 等块存储
 - 优势：性能更好
 - 风险：一次只能被一个 Pod 挂载，如宿主宕机，需等待宿主恢复后才能重新挂载。##### MySQL 操作方式参考：
- 参考 values.yaml 将该部分内容修改至
/usr/local/deepflow/templates/values-custom.yaml 中 mysql 部分：


```
yaml
mysql:
  persistence:      ## 是否启用持久化存储 PVC
  enabled: true      ## 是否使用 storageClass，如需自定义请取消下面注释并修改
                    # storageClass: "xxx"      ## 如想使用已存在的 PVC
                    # existingClaim:
your-claim-pvc-name      accessMode: ReadWriteMany # 访问模式
size: 50Gi # 存储容量
```
- 对至少 2 个以上节点打上 dfdb 标记
- 执行部署流程

Influxdb 操作方式参考：

- 参考 values.yaml 将该部分内容修改至
/usr/local/deepflow/templates/values-custom.yaml 中 influxdb 部分：


```
yaml
influxdb:
  persistence:      ## 是否启用持久化存储 PVC
  enabled: true      ## 是否使用 storageClass，如需自定义请取消下面注释并修改
                    # storageClass: "xxx"      ## 如想使用已存在的 PVC
                    # existingClaim:
your-claim-pvc-name      accessMode: ReadWriteMany # 访问模式
size: 50Gi # 存储容量
```
- 对至少 2 个以上节点打上 dfdb 标记
- 执行部署流程

控制器 HA

依托于 k8s 的自动化调度机制，及受限于 DeepFlow 角色定义，对多个节点标记（label）master_controller 或 slave_controller 即可实现控制器高可用，并只在有 label 的节点上调度。

参考

- [1] 阿里云 NAS 存储卷 https://help.aliyun.com/document_detail/88940.html
- [1] 青云 NAS 存储 <https://docs.qingcloud.com/product/storage/vnas>
- [1] 华为云 NAS 存储 https://support.huaweicloud.com/usermanual-cce/cce_01_0260.html
- [1] 腾讯云 NAS 存储 <https://cloud.tencent.com/document/product/457/44234>

从区域部署

从区域部署 DeepFlow

按照主区域的方法安装 K8s，并完成 DeepFlow 部署（参照基础部署），注意修改主从区域中以下几点内容

从区域配置

- 控制器节点 label 配置为 slave_controller
- 修改 values-custom.yaml 中部署参数
 - node_type=slave
 - 主区域的域名前缀 eg: master-
 - 从区域的域名前缀 eg: slave1-
 - 填写主从区域的控制器 IP 列表
- 一键部署部署从区域 /usr/local/deepflow/bin/df-deploy -i

主区域配置

- 主区域中配置：将 values.yaml 中 customResource 字段内容按注释要求添加至 values-custom.yaml 中
- 主区域中配置：将 customResource.clusterEndpointMasterToSlave.enabled 设置为 true
- 按实际部署情况修改从区域名称及 IP
- 一键更新配置 /usr/local/deepflow/bin/df-deploy -u
- 新增从区域后，需重启主区域 manager, talker, warrant, statistics, pcap 组件

警告： 从区域需要访问主区域 MySQL 数据库、manager、talker，如无法访问需检查以下问题 - 检查负载均衡器端口状态是否为 Healthy kubectl get cep -n deepflow - 节点访问数据库权限 - 防火墙配置问题

防火墙配置参考主从区域通信方式： - master 访问 slave - influxdb: tcp 20044 - genesis: tcp 20015 - elasticsearch: tcp 20042 - 另一部分服务 IP 由服务自行上报，然后直连上报的 IP - clickhouse 9000 - cloud-agent 20230 - pcap-rest 20205 - trisolaris 20014 - slave 访问 master - talker: tcp 20013 - mysql: tcp 30130 - manager: tcp 20403

初始化配置工作

从区域主节点无法访问公网的情形下完成时间同步

- 对于不能访问公网，也没有内部时钟同步源的情形：只能手工修改从区域 k8s master 节点时钟以完成集群时钟同步（K8s 集群部署时所有 node 节点向本集群 master 节点同步时间，master 节点向部署 K8s 时配置的 ntp-server 同步时间）。

所有节点都需要配置

```
timedatectl set-ntp no  
timedatectl set-time "YYYY-MM-DD HH:MM:SS"  
timedatectl set-ntp yes
```

- 有内部时钟源以供同步的情形，需在部署 K8s 时将 ntp-server 配置为内部时钟源，当所有区域的 K8s 时钟源保持一致时，则时间同步一致。

检查集群状态

使用 `kubectl get pods -n deepflow` 检查集群状态。

初始化 Elasticsearch 索引模板

ES 模板会自动初始化，请检查页面告警及采集器日志输出是否正常。

开源模块初始化

简介

登录 DeepFlow 管理页面后，点击 `admin` -> 外部链接可访问各个外部系统

系统	角色
Kibana	以可视化的方式检索 Elasticsearch 数据
Grafana	对 DeepFlow 平台进行的各项性能监控
Cerebro	Elasticsearch 的监控的 GUI 管理界面

Kibana

目前 ES 中存储系统日志、资源变更事件、告警事件，如果需要从 Kibana 中检索这些数据，需要首先初始化索引： - 依次点击 Kibana 页面标题栏 `Management->Index Patterns->Create index pattern` - 填写 Index pattern 为该索引名的正

则表达式 - 若该索引含有时间戳字段的话，下一步可选择 `Time Filter field name` 为该字段

Grafana

初始登录方式 `admin/admin`。

页面定制配置

替换 SSL 证书

k8s 环境中，需先 `kubectl` 工具并通过证书和私钥文件创建 `secret`，并修改 `values-custom.yaml` 中的 `frondEnd.customsSsl.enable=true`

```
kubectl create secret tls cert-deepflow-front-end -n deepflow \
--cert=certs/tls.crt\
--key=certs/tls.key
```

执行如下命令 `kubectl rollout restart ds front-end-daemonset -n deepflow` 重启服务

开启只读管理员类型选项

启用该功能后，创建管理员时会多一个类型选择，普通管理员、只读管理员，使用只读管理员登录后，页面只有查看权限，任何的增删改权限都被取消。通过修改系统->配置管理中 `account->read_only_admin` 值来启用该用户类型。

如果启用了只读管理员，默认只允许 `GET` 请求，如果页面中有用非 `GET` 请求来模拟 `GET` 请求，需要在 `/usr/local/deepflow/templates/front-end/templates/configmap.yaml` 中 `confl.lua` 位置添加该 URL 的白名单，添加示例如下文。

```
WHITE_LIST = {
  {method = "POST", app_uri = "/v1/deepflow/zip"},
  {method = "POST", app_uri = "/deepflow_system_log__0_*/_search"},
  {method = "POST", app_uri = "/v1/rebalance-vtap-analyzer/"},
  {method = "POST", app_uri = "/v1/deepflow/events-aggs/start_time/"},
  {method = "POST", app_uri = "/v1/deepflow/security/analyze/business
_id/"},
  {method = "POST", app_uri = "/rawflow/"},
  {method = "POST", app_uri = "/resource/"},
  {method = "POST", app_uri = "/statistics/"},
  {method = "POST", app_uri = "/analysis/"},
  {method = "POST", app_uri = "/monitor/"},
}
```


登录支持 SSO

DeepFlow 添加支持 sso,多套 DeepFlow 之间可以只登录一台 DeepFlow, 实现其他 DeepFlow 免登录功能。适用于 5.5.3 及以后版本

支持 sso 需要在 系统->配置管理 中 sso 配置项来启用该功能。

```
{
  "sso": {
    "sso_link": "",
    "sso_open": false
  }
}
```

建议: 说明: - sso_open 取值 true 或 false, 默认 false 单点登录关闭, 需要支持 sso 登录的 DeepFlow 必须保证 每台 DeepFlow 中该值一致全是 true 状态 - sso_link 需要支持 sso 登录的其他 DeepFlow 信息, 如果 auth_peer 为 false, 该值不需要配置, 保持默认即可, 自己的 DeepFlow 信息不可以写入自己的 peers_link 中, 可以如下示例设置

示例:

- 如果只有两台 DeepFlow 需要 sso

```
#### DeepFlow_a
name:DeepFlow_A
url:https://12.34.56.78
```

```
#### DeepFlow_b
name:DeepFlow_B
url:https://12.34.56.99
```

DeepFlow_a 中 `sso->sso\_link` 修改配置

```
{
  "sso": {
    "sso_link": "DeepFlow_B,https://12.34.56.99",
    "sso_open": true
  }
}
```

DeepFlow_b 中 `sso->sso\_link` 修改配置

```
{
  "sso": {
    "sso_link": "DeepFlow_A,https://12.34.56.78",
    "sso_open": true
  }
}
```

- 如果三台 DeepFlow 或以上需要 sso

```
#### DeepFlow_a
name:DeepFlow_A
url:https://12.34.56.78
```

```
#### DeepFlow_b
name:DeepFlow_B
url:https://12.34.56.99
```

```
#### DeepFlow_c
name:DeepFlow_C
url:https://12.34.56.00
```

DeepFlow_a 中 `sso->sso\_link` 修改配置

```
{
  "sso": {
    "sso_link": "DeepFlow_B,https://12.34.56.99|DeepFlow_C,https://12.34.56.00",
    "sso_open": true
  }
}
```

DeepFlow_b 中 `sso->sso\_link` 修改配置

```
{
  "sso": {
    "sso_link": "DeepFlow_A,https://12.34.56.78|DeepFlow_C,https://12.34.56.00",
    "sso_open": true
  }
}
```

DeepFlow_c 中 `sso->sso\_link` 修改配置

```
{
  "sso": {
    "sso_link": "DeepFlow_B,https://12.34.56.99|DeepFlow_A,https://12.34.56.78",
    "sso_open": true
  }
}
```

建议： 注意： - sso_link 中的 DeepFlow 名和链接，使用英文中的逗号，来分割，DeepFlow 名可以是中文 - sso_link 中如果有多台 DeepFlow 需要 sso，使用英文中的管道符 | 来分割 - 如果通过 a 用户登录 DeepFlow_a，并 sso 到 DeepFlow_b，如果 DeepFlow_b 中没有用户 a，会创建一个 a 用户，密码默认值为 yunshan1879

单点登录

限制一个账号只能在一个地点，如果在 A 处登录后，再在 B 处登录后，A 处再次请求会提示账户已经在其他地方登录。

配置路径在 系统->配置管理，修改 session->session_single 值为 true

账户安全系数配置

适用于 5.5.7 及后续版本，参考下文的 Web 安全配置

请求 API 的时间查看

可以通过查看某一天的日志来获取后端 app 接口消耗的时间，比如 2020 年 5 月 21 号的日志 /var/log/openresty/app.log.2020-05-21

```
[talker] [21/May/2020:13:57:24 +0800] 153.37.191.130 zhenyu GET 200 0.029s http://172.20.1.128:20013/v1/db-ips/ Request[0]: - Response[8892]: -
```

建议： 说明： - talker: 后端提供改接口的 app 名称 - 21/May/2020:13:57:24 +0800: web 服务器接收到上游返回的数据后，开始写入日志的时间 - 153.37.191.130: 请求方的 ip 地址 - zhenyu: 请求人用户名 - GET: 请求方式 - 200: 接口返回状态码 - 0.029s: web 服务器和上游服务建立连接到接收完最后一个字节消耗的时间 http://172.20.1.128:20013/v1/db-ips/: 请求的 api

获取永久有效的 access_token

适用于 5.5.7 及后续版本

通过 curl 执行如下命令 curl

http://127.0.0.1:20822/valid_token/<email>/<pwd>，得到如下结果

```
{
  "OPT_STATUS": "SUCCESS",
  "DATA": "gAAAAABexhfQWERDQWESDFShVjc1JhRDNxbsF2btXVND-i8Qbo_xAqe0IQ6N
KJaH13Yg9MmlwoDbYn0tI7Mzpnjdqgv1iR_gKQhNCdaF01d3w2YkSg_ZaT8rMpXEN82oSnI
-MaHU4iUFRpgc9MAs_fi2dfs9bMtYZw04MMkAsDF6LZv4u6nvc07pIrFGDi0KaBYd5A71r8
wHfryXyAQWERAfQDSDFQW5w_ZLEHGg==",
  "DESCRIPTION": "SUCCESS"
}
```

DATA 值为永久有效的 access_token 值

建议： 说明：请求的 url 中的 <email> 和 <pwd> 请换成需要获取永久有效 token 的账户邮箱和密码

DeepFlow 界面定制

界面配置文件位置

/usr/local/deepflow/templates/front-end/templates/configmap.yaml 中的 app_config_user.js

/usr/local/deepflow/templates/front-end/templates/configmap.yaml 中的 app_config.js

建议： 注意： - 其中 app_config.js 是默认配置文件，app_config_user.js 是用户配置文件； - 为了使配置不受影响，请修改 app_config_user.js 中的配置 - 用户同时修改 app_config_user.js 与 app_config.js 时，app_config_user.js 中的配置会优先生效。 - 升级时会覆盖 app_config.js，不会覆盖 app_config_user.js

安装模式控制

通过编辑配置文件中的 INCLUDE_APPS 字段的值来切换安装模式。目前只支持 npm 和 npb。

全景图多选框上限控制

修改配置文件中的 FLOW_VIEW_MULTIPLE_SELECT_LEN 字段来修改多选框的上限个数。

全景图资源集合上限控制

修改配置文件中的 FLOW_VIEW_MAX_RESOURCE_SET 字段来修改资源集合的上限个数。

全景图折线图自定义展示个数上限控制

修改配置文件中的 FLOW_VIEW_CUSTOM_LINE_CHART_NUM 字段来修改折线图自定义展示上限个数。

子视图中数据源秒级粒度选择时间范围

修改配置文件中的 SUPPORT_SECOND_DATA_TIME 字段，当选择的时间范围开始时间到当前时刻大于该范围时，不可选择秒级粒度。

页面菜单控制

- 5.7.1（包括 5.7.1）以后版本配置菜单显示如下：
 - 在 app_config_user.js 中增加 HIDDEN_MENU_PATH_LIST:[] 配置，并将需要隐藏的路径填入该字段中，隐藏单个页面的路径为浏览器中显示该页面时 url 中#后面，? 前面的部分，
 - 例如需要隐藏流量搜索页面，则配置 HIDDEN_MENU_PATH_LIST:['/platformOverview/flowView']
 - 如果隐藏整个模块页面，例如分发的所有页面，则配置 HIDDEN_MENU_PATH_LIST: ['/npb']

- 对于从 5.7.1 之前的版本升级到 5.7.1 的，如有需要隐藏的菜单请在 `app_config_user.js` 配置 `HIDDEN_MENU_PATH_LIST` 字段 ##### 浏览器标签栏图标
- 修改标签栏图标
 - 将 `/var/www/lcweb/public/images/favicon.ico` 替换。favicon 尺寸为 32*32 像素。

平台信息内厂商信息、服务邮箱

将 `/usr/local/deepflow/templates/rear-end/templates/configmap.yaml` 中 `deepflow.conf` 对应的选项修改后更新进 k8s 即可。

```
[global]
version = DeepFlow-5.1
company = 北京云杉世纪网络科技有限公司
support_email = noc@yunshan.net.cn
```

版本信息由打包工具生成，普通情况下应禁止修改。

平台外部链接信息

由 `/usr/local/deepflow/templates/rear-end/templates/configmap.yaml` 中 `deepflow.conf` 中下面的选项设定。

```
[deepflow-web]
outer_link.kibana = /kibana/
outer_link.grafana-server = /grafana-server/
outer_link.cerebro = /cerebro/
outer_link.jstorm = /jstorm/
```

版本信息由打包工具生成，普通情况下应禁止修改。

视图创建数量配置

将 `/etc/web/df-web.yaml` 中对应的选项修改，然后重启 `df-web-service` 服务。

```
view:
  admin_screen_limit: 10      # 管理员创建视图数量
  tenant_screen_limit: 10     # 租户创建视图数量
  admin_dashboard_limit: 15   # 管理员创建子视图数量
  tenant_dashboard_limit: 15  # 租户创建子视图数量
```

Web 安全配置

- 过滤请求方式

系统默认允许 GET|POST|PUT|PATCH|DELETE|OPTIONS 请求，通过修改 /usr/local/openresty/nginx/lua/conf/user.lua 中的 SAFE_METHOD 值为 true 并重启 openresty 会禁止如下请求 PUT|PATCH|DELETE。

- 开启接口防篡改和重放攻击

修改 /usr/local/openresty/nginx/lua/conf/user.lua 文件中 CRYPTO_OPEN 值为 true 来开启该功能，默认关闭（false）。然后重启 openresty。

为了防止客户端时间和服务器时间两者有少许的误差导致校验失败，可以修改该文件下的 CRYPTO_DEVIATION_TIME 值，默认是 3 秒的误差（客户端时间可以比服务器时间大 3 秒）。

可以修改该值，但是不允许该值大于 60 秒，如果客户端时间和服务器时间相差太大超过 60 秒，建议使用北京时间为标准来校对两者的时间。

- 通过 系统->配置管理 来配置安全参数，默认配置内容如下

```
{
  "sso": {
    "sso_link": "",
    "sso_open": false
  },
  "account": {
    "account_allow_login_white_list_ip": "*",
    "account_allowed_login_max_time": 0,
    "account_allowed_login_min_time": 0,
    "account_allowed_login_time_period": false,
    "account_first_login_change_pwd": false,
    "account_login_failed_count": 0,
    "account_login_failed_locked_time": 60,
    "account_not_change_pwd_lock_time": 0,
    "account_not_login_lock_time": 0,
    "account_second_check": false,
    "read_only_admin": false,
    "use_ungrouped_type": false,
    "user_limit": 100,
    "user_tenant_limit": 200,
    "verifycode_use": false,
    "radius_account_switch": false
  },
  "password": {
    "pwd_include_case": false,
    "pwd_include_number": false,
    "pwd_include_special_chars": false,
    "pwd_include_string": false,
    "pwd_max_len": 16,
    "pwd_min_len": 8
  },
  "session": {
```

```

    "session_inactive_close": false,
    "session_inactive_close_interval": 0,
    "session_max_online": 0,
    "session_one_client": false,
    "session_single": false
  },
  "file_storage": {
    "file_storage_extension": "rpm,iso,gz,zip,tar",
    "file_storage_size": 3072,
    "file_total_size_max": 51200,
    "file_count_max": 100,
    "file_storage_support_admin": false
  }
}

```

- 配置参数说明如下

sso:

```

# 多控制器单点登录, false(默认不开启)
sso_open: false
# 需要多控制器单点登录的ip 或 domain, 除了当前环境
# 默认""
# 只有一个控制器示例: "baidu,http://baidu.com"
# 多个控制器示例: "baidu,http://baidu.com|qq,http://qq.com|google,http://google.com"
sso_link: ""

```

password:

```

# 密码最短(系统允许取值的最小值)
password_min_len: 8
# 密码最长(系统允许取值的最大值)
password_max_len: 32
# 包含数字。false(默认不校验)
password_include_number: false
# 包含字符串。false(默认不校验)
password_include_string: false
# 包含大小写。false(默认不校验)
password_include_case: false
# 包含特殊字符。false(默认不校验)
password_include_special_chars: false

```

account:

```

# 允许登录时间段。false(默认关闭)。时间段取值范围(0-23)
account_allowed_login_time_period: false
account_allowed_login_min_time: 0
account_allowed_login_max_time: 0
# 多长时间未登录, 锁定账户。单位/天。0 值不开启(默认), 取值范围(0-365)
account_not_login_lock_time: 0
# 多长时间未修改密码, 锁定账户。单位/天。0 值不开启(默认), 取值范围(0-365)

```

```
account_not_change_pwd_lock_time: 0
# 首次登录需要修改密码。false 不开启(默认)
account_first_login_change_pwd: false
# 用户模块开启二次校验。false 不开启(默认)
account_second_check: false
# 允许登录失败次数。0 值不开启(默认), 取值范围 (0-10)
account_login_failed_count: 0
# 达到登录失败后, 锁定时间。单位/秒, 取值范围 (60-600)
account_login_failed_locked_time: 60
# 允许登录的客户端(ip), "*"默认不限制, 限制格式如:"127.0.0.1,192.168.1.1",
允许最多单点登录的控制器数量 (1-5)
account_allow_login_white_list_ip: "*"
# 登录使用验证码。false 关闭(默认)
verifycode_use: false
# 登录开启授权校验。true 开启(默认)
# verify_license: True
# 管理员账户数量。取值 (1-100)
user_limit: 100
# 租户账户数量。取值 (1-200)
user_tenant_limit: 200
# 开启未分组用户类型 (101)。false 不开启(默认)
use_ungrouped_type: false
# 开启只读管理员选项。false 不开启(默认)
read_only_admin: false
# 支持用radius 认证账户登录系统。false 不开启(默认)
radius_account_switch: true
```

session:

```
# 非活动的会话终止时间。false(默认不开启)。单位/分钟, 取值范围 (0-30)
session_inactive_close: false
session_inactive_close_interval: 0
# 互踢下线(一个账户只能在一个地方登录)。false(默认不开启)
session_single: false
# 会话最大在线人数。0 值(默认不限制), 取值范围 (0-100)
session_max_online: 0
# 一个客户机只能登录一个账号。false(默认不开启)
session_one_client: false
```

file_storage:

```
# 支持上传的文件后缀, 值等于 "*" 表示不判断文件后缀
file_storage_extension: "rpm,iso,gz,zip,tar"
# 单个文件最大, 单位/M, (系统允许的最大值 3072M)
file_storage_size: 3072
# 已经上传的文件总大小, 单位/M
file_total_size_max: 51200
# 允许上传文件最大数量
file_count_max: 100
```


是否允许普通管理员使用该功能（超级管理员默认有权限），*false*(默认不开启)。
`file_storage_support_admin: false`

部署限制

ARM 环境部署限制

- kibana 没有合适版本，kibana 工具无法使用
- apm-server 没有合适版本，相关功能暂无法使用
- baseline 没有对应依赖库，相关功能暂无法使用
- 安装若出现 apm-server, baseline 安装失败，注释掉对应 salt 代码重新安装
- trident 暂不提供 debian/ubuntu 安装包，需要直接通过 trident 二进制文件部署
- procsnoopd 不支持，无法监控进程重启

K8s 环境部署限制

- 不支持主备控制器切换操作，所有控制器均为无状态
- 不支持控制器配置文件导出
- 不支持一键升级，需要上传所有镜像、重建 POD
- 进程配置文件修改后，需要重建 POD 才能生效
- procsnoopd 不支持，无法监控进程重启
- HA 需要环境有共享存储（NAS），否则不支持 HA

IPv6 环境部署限制

- Kibana 在 IPv6 环境下无法创建 Index

Kubernetes 部署模式

已全面升级为 K8s 部署模式，文档已同步至【基础部署】此页将于 5.7.4 版本移除

虚拟机自动部署模式

适用场景

阿里公有云上支持通过预定义的虚拟机镜像和 `user-data`（实例自定义数据）实现 DeepFlow 的一键部署。

虚拟机镜像

- 控制器和数据节点使用如下镜像
 - <https://ali-deepflow-200g.oss-cn-beijing.aliyuncs.com/deepflow-offline-release-R5.6.3-217-2.qcow2>
 - 版本：v5.6.3-R3 + 13 号 Patch

控制器/数据节点创建

- 主控制器创建

- 注意: 目前仅支持创建一个主控制器
- 创建时携带如下固定的 user-data (实例自定义数据)

```
#!/bin/sh
sed -i '/deepflow/d' /etc/hosts
rm -rf /root/.pip
df-init 127.0.0.1 controller,master,elasticsearch-warm >> /var/log/df-init.log 2>&1
```

- 从控制器创建 (可选)

- 注意: 需要提前获知主控制器的 IP 地址
- 创建时携带如下固定的 user-data (实例自定义数据)

```
#!/bin/sh
sed -i '/deepflow/d' /etc/hosts
rm -rf /root/.pip
df-init <主控制器 IP 地址> proxy_controller >> /var/log/df-init.log 2>&1
```

- 数据节点创建

- 注意: 需要提前获知主控制器的 IP 地址
- 创建时携带如下固定的 user-data (实例自定义数据)

```
#!/bin/sh
sed -i '/deepflow/d' /etc/hosts
rm -rf /root/.pip
df-init <主控制器 IP 地址> tsdb,elasticsearch-warm >> /var/log/df-init.log 2>&1
```

- 部署检查

- 查看/var/log/df-init.log, 检查 df-init 脚本的执行情况和进度
- 检查 df-cloud-init 进程是否正常启动

systemctl status df-cloud-init
- 查看/var/log/df-cloud-init.log, 检查部署进度

镜像更新

- 更新/root/目录下的 ISO
- 修改/etc/fstab 的挂载配置, 指向更新后的 ISO

数据节点使用对象存储

数据节点使用对象存储

- 对象存储接口需支持 AWS 的 S3 接口协议
 - 如: 使用百度的对象存储 BOS 的 S3 接口参考:
<https://cloud.baidu.com/doc/BOS/s/Fjwvyq9xo>
- 启用对象存储
 - 配置/usr/local/deepflow/templates/values-custom.yaml 的 global
 - s3StorageEnabled 设置为 true
- ClickHouse 对象存储配置
 - 配置/usr/local/deepflow/templates/values-custom.yaml 的 clickhouse 下的 s3StorageConfig
 - endpoint: 配置对象存储的 S3 bucket URL。格式:
`http://<domain>/<bucket>/<path>`
 - access_key_id: 对象存储访问密钥 ID
 - secret_access_key: 对象存储的密钥 ID
 - 配置示例:

```
clickhouse:
  s3StorageConfig:
    endpoint: http://s3.bj.bcebos.com/bucket-xx/host/
    accessKeyId: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
    secretAccessKey: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```
- 配置对象存储保存时长
 - 在 droplet 和 clickhouse 都启动成功后
 - 然后在 WEB 管理页面系统->数据节点->监控数据存储配置->数据源列表->操作, 重新设置保存时长.
 - 注意: 该保存时长是本地数据的保存时长, 对象存储中数据的保存时长,是本地数据保存时长的 3 倍

采集器部署

简介

本章节描述 DeepFlow 采集器 (Trident) 的安装方法。

采集器部署

具体操作详见各个子章节。

警告： 已发现在腾讯公有云上云平台会将云服务器发送的超过 1400B 的 UDP 包修改末尾的几十字节；在阿里公有云上云平台会限制云服务器发 1400B~1500B 之间的 UDP 包。因此，建议在公有云中部署采集器时，将裸 UDP 最大 MTU 设置为 1400 字节。

建议： 当部署环境中存在外部负载均衡器为采集器提供服务时，可将控制器和数据节点的 NAT IP 配置为负载均衡器的 VIP 进行适配。但如果不同的采集器使用不同的 VIP（例如阿里专有云场景），可以在 trident 配置文件中指定 analyzer-ip 为数据节点的 VIP，此时采集器也会忽略控制器下发的控制器 IP，永远使用配置文件中的第一个 controller-ips 请求控制器。

建议： 采集器配置文件中的 controller-ips 支持填写域名，目前在 SaaS 场景下使用此特性。

采集器注册

采集器部署完成后会自动向控制器请求注册。如果关闭了自动注册机制（默认未关闭），则需要管理员在系统-采集器-列表页面手动完成注册操作。

采集器被自动发现后会直接加入默认采集器组，若希望修改此行为，可通过在 Trident 的 yaml 配置文件中将 vtap-group-id-request 设置为系统-采集器-组页面的某个 ID 字段值，使得采集器被发现时可自动加入特定的采集器组。提示：该配置仅影响初始加入的采集器组，管理员后续同样也能在页面修改所属采集器组。

KVM、Workload 和容器类型采集器远程升级

本文档描述了在容器化部署模式下主控制器、从控制器上远程升级 trident 的方式：

- 检查主控制器、从控制器上是否已安装最新版本的 trident；若未安装，则使用如下命令安装，注意替换对应版本 rpm 包：

```
yum --disablerepo=* localinstall /usr/local/deepflow/extra-artifacts/trident-installation-package/x86_64/trident-1.5-1363.el7.x86_64.rpm
```

- 执行 trident -v 获取版本号

```
[root@master1 ~]# trident -v
1363-148ac5f14aef81c98470fd30468bd8de86366730 2021-10-29
# 示例，revision 为 `1363-148ac5f14aef81c98470fd30468bd8de86366730`
```

- 修改主控制器上 /usr/local/deepflow/templates/values-custom.yaml，将 trident_revision 修改为 trident 的 revision
- Windows 环境 trident 升级还需执行如下步骤：
 - 注意：原版本和升级目标版本所对应的 DeepFlow 版本都不能低于 5.5.8

- 安装 trident.exe 到主备从控制器，注意替换对应版本 rpm 包：

```
yum --disablerepo=* localinstall /usr/local/deepflow/extra-artifacts/trident-installation-package/x86_64/trident-windows-1.5-1360.el7.x86_64.rpm
```
- 验证 windows 和 linux 的版本保持一致：yum --disablerepo=* list history trident-windows trident
- 容器采集器的升级还需要参考容器采集器部署的章节，将镜像上传到仓库（或 node 本地）并打上 latest 标签
- 执行命令应用上述修改： /usr/local/deepflow/bin/deepflow-deploy -uo trisolaris
- 进入 mt 容器，在 mntnct 容器中执行 mt 相关命令：

```
MT_POD=$(kubectl get pods -n deepflow | grep mntnct | awk '{print $1}')
```

```
kubectl exec -it $MT_POD -n deepflow sh
```
- 在容器中执行 mt vtap.upgrade name=XXX 升级对应的 vtap

采集器批量部署

采集器批量部署介绍

- 使用 Ansible playbook 部署
- 支持使用 Yum/Apt 作为包管理器的 systemd 环境系统
- 需要在 Nginx 中维护 Yum/Apt 源和相关 repo、source.list 文件
- 需要在 Nginx 中维护采集器配置

Ansible 批量部署

主要为两个文件

1. inventory 文件：保存 ansible 的 hosts 和环境变量，需要根据不同的环境做不同配置
2. playbook.yaml 文件：采集器批量部署的 ansible playbook

Ansible 版本要求：2.7+

inventory

[trident]

需要部署采集器的 IP 列表

需要 root 账户或者具有 root 权限的 sudo 账户

如果主机的 ssh 端口、用户、密码不一样时，可以配置指定 host 的配置

```
#### 172.20.0.21 ansible_port=22  ansible_user=root ansible_ssh_pass=Pa  
ssword ansible_sudo_pass=Password  
172.20.0.23  
172.20.0.22  
172.20.0.24
```

[all:vars]

#####需要部署采集器主机的端口

ansible_port=22

#####部署采集器使用的账户

ansible_user=root

#####部署采集器使用的密码，和密钥二选一

ansible_ssh_pass=Password

####使用 sudo 账户安装时，需要指定此密码，一般为 ssh 密码，这个密码和 ssh 密码/
密钥是共存的

#####ansible_sudo_pass=Password

#####部署采集器使用的密钥，和密码二选一

###ansible_ssh_private_key_file=/config/id_rsa

#####DeepFlow source IP

####deepflow nginx 的IP 端口，后台会自动+http

deepflow_source_address=10.33.40.26:8080

####安装Yum/Apt 包的重试次数，默认即可，无需修改

pkg_install_retries=4

retry_stagger=5

playbook.yaml

```
- hosts: trident  
  tasks:  
    - name: Fetch /etc/os-release  
      raw: cat /etc/os-release  
      register: os_release  
      changed_when: false  
      check_mode: false  
      become: true  
  
    - name: Configure trident repository on apt package manager  
      get_url:  
        url: "http://{ deepflow_source_address }}/repo/deepflow-trident.list"  
        dest: "/etc/apt/sources.list.d/deepflow-trident.list"  
        mode: 0644  
        when: ansible_pkg_mgr == 'apt'
```

```

    become: true

- name: Update apt repository cache
  apt:
    update_cache: yes
  when: ansible_pkg_mgr == 'apt'
  become: true

- name: Configure trident repository on yum package manager
  get_url:
    url: "http://{{ deepflow_source_address }}/repo/deepflow-trident.repo"
    dest: "/etc/yum.repos.d/deepflow-trident.repo"
    mode: 0644
  when:
    - ansible_os_family == "RedHat"
  become: true

- name: Install trident packages requirements
  package:
    name: "trident"
    state: present
  register: extra_pkgs_task_result
  until: extra_pkgs_task_result is succeeded
  retries: "{{ pkg_install_retries }}"
  delay: "{{ retry_stagger | random + 3 }}"
  become: true

- name: Configure trident config file
  get_url:
    url: "http://{{ deepflow_source_address }}/config/trident.yaml"
    dest: "/etc/trident.yaml"
    mode: 0644
  become: true

- name: trident | restart trident service
  service:
    name: "trident"
    state: restarted
    enabled: yes
  become: true

```

执行 playbook

```
ansible-playbook --become --become-user=root -i inventory playbook.yaml
```

制作 Yum/Apt 源

建议将 Yum/Apt 源放在 DeepFlow 控制器上,整体目录结构如下

####进入nginx的root目录
cd /opt/kubeplay/resources/nginx/
#####创建相关目录, trident 为采集器包存放目录, repo 为存放Yum
###m的repo文件和apt的source.list文件目录, config为存放采集器配置文件目录

```
mkdir -p trident repo config
```

#####如下为目录结构

```
tree ./trident ./repo ./config/
```

#####trident rpm、deb包存放目录

```
./trident
├── repodata
├── trident-1.5-1379.el7.x86_64.rpm
└── trident-1.5-1379.systemd.amd64.deb
./repo
├── deepflow-trident.list
└── deepflow-trident.repo
./config/
└── trident.yaml
```

制作Yum/Apt源方法

#####需要提前安装 createrepo 和 dpkg-scanpackages 工具制作Aum/Apt源
#####此步骤操作可以在外面执行, 做好rpm/apt源后将整个目录打成tar包, 到现场解压。

```
createrepo -d /opt/kubeplay/resources/nginx/trident/
cd /opt/kubeplay/resources/nginx/
dpkg-scanpackages trident/ | gzip -9c > trident/Packages.gz
```

Yum repo 文件: repo/deepflow-trident.repo

```
[deepflow-trident]
name=deepflow-trident
#####repo文件放在deepflow的Yum源中, 由deepflow管理, 下面的IP端口填写deepflow Yum/Apt源的IP和nginx端口
baseurl=http://10.33.40.26:8080/trident/
enabled=1
gpgcheck=0
sslverify=0
```

Apt source.list 文件: repo/deepflow-trident.list

```
#####repo文件放在deepflow的Yum源中, 由deepflow管理, 下面的IP端口填写deepflow Yum/Apt源的IP和nginx端口
deb [trusted=yes] http://10.33.40.26:8080/ trident/
```

采集器配置路径: 现场根据自身情况配置, 注意 controllerIP 地址需要写到该文件中, 部署采集器时会下载该文件保存到/etc/trident.yaml 并直接启动采集器

config/trident.yaml

安全传输 [可选]

基于安全的考虑，采集器与控制器之间的控制信令可使用 TLS 协议，实现 Trident 对控制器的身份认证，控制面数据的加密传输。

首次配置

采集器默认未开启安全通信，如需开启，需在采集器上安装控制器证书，实现对控制器的可信认证和安全通信，步骤如下： - （vSphere 环境除外）请手动将从控制器证书/etc/pki/tls/certs/cert.csr.\$控制器控制 IP 地址拷贝到采集器所运行操作系统的/etc/目录下 - 注意将文件重命名为 trident_controller.cert.\$控制器控制 IP 地址 - 其中\$控制器控制 IP 地址就是 trident.yaml 中的 controller-ips，若多个 IP 需要拷贝多个证书 - 修改 trident.yaml: `yaml controller-cert-file-prefix: /etc/trident_controller.cert` - 重启 trident - 举例如下： - 假设证书放在采集器的/etc 下，Trident 配置的从控制器 IP 为 10.10.10.10、10.10.10.11 - 那么分别修改证书名为 trident_controller.cert.10.10.10.10、trident_controller.cert.10.10.10.11 - 设置证书路径前缀为 controller-cert-file-prefix: /etc/trident_controller.cert

建议： - 对于 vSphere 环境，部署时已安装证书，开启安全通信，仅需修改配置文件，并重启 trident 即可 - 安全通信使用的默认端口是 20135

证书存放路径： - 控制器证书存放路径相同 - 私钥：
/etc/pki/tls/private/private.pem.\$控制器控制 IP 地址` - 证书：
/etc/pki/tls/certs/cert.csr.\$控制器控制 IP 地址``

更新证书

当需要更新证书时，可使用 `mt saltstack.gen_cert` 命令重新生成所有控制器上的证书： - `sync`：可选同步控制器证书到控制器、采集器；`yes` 表示同步，`no` 表示不同步，默认值：`yes` - 同步仅针对 vSphere 环境，对于其他环境需参考首次配置的方法手动更新 - 证书更新后，请重启 trisolaris 和 trident，证书方可生效

KVM

依赖检查

原理上 Trident 仅依赖于 Linux Kernel 2.6.23+。

采集配置

trident.yaml 配置文件： - 推荐：将 `vtap-group-id-request` 设置为系统-采集器-组页面的某个 ID 字段值，使得采集器被发现时可自动加入特定的采集器组

系统-采集器-配置页面： - 采集网口配置为采集流量的网口的正则表达式，请添加^强制匹配开头，例如：`^(eth|tap).*-eth.*`：采集 KVM 出接口（物理网卡）的流量，以展现虚拟网络链路追踪能力 - 青云私有云注意事项：当 KVM 使用了硬件网

卡直通或 VXLAN offloading 能力时，除了出接口（仍然会有部分流量无法 offload）以外，需要额外配置以下接口 - **vg_tep_out\$**：非直通网卡模式需要配置，在 bridge **br_vg_out** 中，网卡名为 MAC 地址后 4B 的是 LB 的虚接口（背后是一个 lxc container） - **vf.***：连到同一个子网的虚拟机网卡（名为 MAC 地址后 4B）在一个 bridge 中，这个 bridge 中 vf 前缀的虚接口负责调用硬件网卡的 vxlan 封装能力，这个网口可认为是该 bridge 的出接口。这个接口的 vf 前缀后会跟着物理网卡名称的前缀，例如 **vfennp** 或者 **vfenns - [0-9a-zA-Z]+_[0-9]{1,3}\$**：青云直通网卡场景需要配置该接口 - **tap.***：采集虚拟机的虚拟网卡的流量 - 注意：采集网口配置必须遵循只看两端不看中间的原则，仅采集虚拟机、物理交换机进出宿主机的流量，请勿配置除此之外的其他网口，否则会导致采集压力上升，并导致页面显示数据错误。 - 虚拟机 MAC 解析默认配置为接口 MAC 即可，具体候选项含义为 - 接口 MAC：适用于大多数场景，通过 tap 口的 MAC 后 5 字节确定虚拟机 MAC，第一字节忽略 - 接口名称：Contrail 采用此方式，tap 口的名称后缀代表 MAC 地址的后 5 字节，同样第一字节忽略 - 早期版本的青云私有云也要使用此方式，且只能获取到 MAC 地址后 4 字节。但是 2020 年接触的青云私有云已无此问题，直接可使用接口 MAC - 青云环境如果使用直通网卡，仍然需要配置为接口名称来通过直通网卡的名称查询 bridge fdb 表获取 MAC 地址 - 虚拟机 XML：在某些版本华为私有云中，tap 口的 MAC 和名称均和虚拟机实际 MAC 无关联，此时需要读取虚拟机的 QEMU XML 文件确定实际 MAC - 隧道解封配置为正确的 SDN 隧道协议，以展示虚拟网络链路追踪能力 - 当 KVM 上使用 VXLAN 隧道封装时 - 若未部署 KVM 上虚拟机内的容器采集器，或容器跨节点通信不使用隧道封装，需要配置 KVM 采集器隧道解封参数为 VXLAN - 若 KVM 上虚拟机运行容器、部署了容器采集器、且容器跨节点通信使用隧道时，需根据容器 CNI 隧道类型配置 KVM 采集器的隧道解封参数为 VXLAN-VXLAN 或 VXLAN-IPIP - 对于腾讯专有云（TCE）场景，需要配置为 TCE GRE

安装 Trident

一般 KVM 环境需要客户自行将 trident 的 RPM 安装到宿主机上。包含如下步骤：

将采集器 rpm 包拷贝至计算节点/tmp 目录下并安装

```
yum --disablerepo=* localinstall -y /tmp/trident-*.rpm
```

修改/etc/trident.yaml 文件中的 controller_ips，此为 DeepFlow 主备控制器 IP

```
sed -ir 's#127.0.0.1#10.10.241.115#g' /etc/trident.yaml
```

按需修改 trident.yaml 中的 vtap-group-id-request

启动 trident 服务并设置 trident 服务开机启动

```
systemctl enable trident && systemctl start trident
```

检查服务正常运行

```
systemctl status trident
```

杀死 trident 并继续检查是否可恢复运行

```
pkill -9 trident
systemctl status trident
```

KVM-DPDK

依赖检查

建议单独隔离出来一个 CPU（隔离以及检查方法见下文）给 trident 收包使用，trident 的 CPU 限制不小于 2：

警告： 注意：核隔离仅为提升采集性能，不是必须的。应该由客户提前配置好，并告诉我们 trident 使用哪个核。

CentOS 上配置核隔离参考方法，以隔离 CPU6 为例：

step 1: 在 grub 文件里增加 isolcpus 配置项

```
vi /etc/default/grub
GRUB_CMDLINE_LINUX="isolcpus=6"
```

step 2: 重启使配置生效

```
grub2-mkconfig -o /boot/grub2/grub.cfg
reboot
```

step3: 重启完后，再确认一下配置

```
cat /proc/cmdline
BOOT_IMAGE= ... isolcpus=6
```

采集配置

采集器 trident.yaml 配置： - 通过 dpdk-ring-port 指定流量采集网口，默认为 dpdkr0 - 推荐：将 vtap-group-id-request 设置为系统-采集器-组页面的某个 ID 字段值，使得采集器被发现时可自动加入特定的采集器组

安装 Trident

参照 KVM 环境的安装方法完成 trident rpm 的安装。在此基础上的余下工作现阶段需要到采集结点进行手动配置，以后会实现自动化配置：

创建 dpdkr 口，这里命名为 dpdkr0

```
DPDK_RING_PORT=dpdkr0
ovs-vsctl add-port br-int $DPDK_RING_PORT \
  -- set interface $DPDK_RING_PORT type=dpdkr mtu_request=9000
```

配置 ovs 流量镜像策略

```
ovs-vsctl -- --id=@p get port $DPDK_RING_PORT \
  -- --id=@m create mirror name=m0 select_all=true output_port=@p \
  -- set bridge br-int mirrors=@m
```

修改 trident.yaml 并重启 trident 服务：

```

tap-mode: 1                # 设置为mirror 模式
ovs-dpdk-enable: true      # 使能DPDK，默认为false
dpdk-pmd-core-id: 6        # 隔离出来给 trident 使用的收包 CPU
dpdk-ring-port: "dpdkr0"   # trident 从 ovs-dpdk 的哪个口上收包，即上文中的$D
                             PDK_RING_PORT

```

实际部署举例

给 Trident 隔离一个 CPU [可选]

- 将 /etc/nova/nova.conf 的配置项 vcpu_pin_set 从

```
vcpu_pin_set=0-2,9,20-22,29,10-19,30-39
```

修改为如下（即删掉 CPU-#9，后面会用于 Trident，该参数的意思是这些 CPU 可以被 VNF 分配，除 VNF 外的普通 VM 也不会用到）

```
vcpu_pin_set=0-2,20-22,10-19,30-39
```

- 将 /etc/default/grub/cmdline 的配置项 GRUB_CMDLINE_LINUX 从

```
GRUB_CMDLINE_LINUX="crashkernel=512M rhgb quiet LANG=en_US.UTF-8 de
fault_hugepagesz=1G hugepagesz=1G hugepages=120 intel_iommu=on iomm
u=pt skew_tick=1 skew_tick=1 isolcpus=3-8,23-28 intel_pstate=disabl
e nosoftlockup nohz=on nohz_full=3-8,23-28 rcu_nocbs=3-8,23-28 l1tf
=flush kvm-intel.vmentry_l1d_flush=always"
```

修改为如下（即将 nohz_full 和 rcu_nocbs 修改为包含 9，说明进程调度部要使用 CPU-#9）

```
GRUB_CMDLINE_LINUX="crashkernel=512M rhgb quiet LANG=en_US.UTF-8 de
fault_hugepagesz=1G hugepagesz=1G hugepages=120 intel_iommu=on iomm
u=pt skew_tick=1 skew_tick=1 isolcpus=3-9,23-29 intel_pstate=disabl
e nosoftlockup nohz=on nohz_full=3-9,23-29 rcu_nocbs=3-9,23-29 l1tf
=flush kvm-intel.vmentry_l1d_flush=always"
```

- 执行命令

```
# 让上面的grub 参数写入 /boot/grub2/grub.cfg 配置文件，重启生效
grub2-mkconfig -o /boot/grub2/grub.cfg
```

- 执行命令 reboot
- 执行命令计算节点上增加一条静态路由：

```
# 这是客户增加的临时静态路由，目的是跟控制器网络通
route add -net 10.135.127.0/24 gw 10.135.124.1
```

配置 Ovs 镜像

- 执行命令

```
# 创建一个名为 dpdkr0 的 ring 口，用于 Trident 采集流量
ovs-vsctl add-port br-int dpdkr0 -- set interface dpdkr0 type=dpdkr
mtu_request=9000
```

- 执行命令

```
# 将 OVS 上所有虚接口的流量镜像到刚才创建的 dpdkr0 口
ovs-vsctl -- --id=@p get port dpdkr0 \
-- --id=@m create mirror name=m0 select_all=true output_port=@p \
-- set bridge br-int mirrors=@m
```

安装 Trident

- 使用 yum 或 rpm 命令正常安装 trident rpm 包
- 修改 trident.yaml 的如下配置项：

```
controller-ips:
  - 10.135.127.6      # 填写 DeepFlow 控制器 IP 地址
tap-mode: 1          # DPDK 环境必须设置为 1，表示 MIRROR
ovs-dpdk-enable: true # DPDK 环境必须设置为 true
dpdk-pmd-core-id: 9  # 可选项，填写 Trident 使用的 CPU
```

- 执行命令启动 trident:

```
systemctl enable trident
systemctl start trident
```

Hyper-V

依赖检查

检查宿主机的 C:\Windows\System32 目录下是否有 wpcap.dll，否则需要安装 wpcap.dll，推荐使用 [Win10Pcap](#)。检查宿主机是否安装 npf 服务，使用命令 `sc query npf`

警告： 目前不支持同一个宿主机上虚拟机 IP 地址重叠的场景

采集配置

采集器 trident.yaml 配置： - 推荐：将 `vtap-group-id-request` 设置为系统-采集器-组页面的某个 ID 字段值，使得采集器被发现时可自动加入特定的采集器组

系统-采集器-配置页面： - 隧道解封配置为正确的 SDN 隧道协议，以展示虚拟网络链路追踪能力 - 当 Hyper-V 上使用 VXLAN 隧道封装时 - 若未部署 Hyper-V 上虚拟机内的容器采集器，或容器跨节点通信不使用隧道封装，需要配置 Hyper-V 采集器隧道解封参数为 VXLAN - 若 Hyper-V 上虚拟机运行容器、部署了容器采集器、且容器跨节点通信使用隧道时，需根据容器 CNI 隧道类型配置 Hyper-V 采集器的隧道解封参数为 VXLAN-VXLAN 或 VXLAN-IPIP - 采集网口配置包含宿主机业务网卡

的正则表达式，trident 会采集该接口的流量 - 注意：不要采集配置有 IP 地址（管理口等）的网卡，这部分流量采集后全景图中的统计位置会展示为客户端宿主机或服务端宿主机（实际上展示为客户端或服务端更合理） - 也可通过 trident-windows.yaml 中的 src-interfaces 配置精确的采集网口

安装 Trident

- 按照 Workload-V 中的方法安装 Trident
- 修改 trident-windows.yaml 配置文件
 - tap-mode 修改为 mirror 模式
- 重启 trident 服务

ESXi

依赖检查

每个宿主机需要部署一台虚拟机，作为 Node 节点加入 K8s 集群，并通过在 vSphere 中配置本地 SPAN 将流量镜像给采集器虚拟机。

- 创建采集器虚拟机，确保每个计算节点都有一台虚拟机
- 扩容虚拟机为控制器所在 K8s 集群 Node 节点，参考文档《部署扩容-控制器与数据节点部署-准备工作-4.2.4.基本操作-扩容节点》
- 检查采集器节点与控制器、数据节点控制平面的连通性

警告： 采集器虚拟机必须安装 VMware-Tools，虚拟网卡应当选择 vmxnet3 类型，部署前请务必确认。

采集配置

修改 values-custom.yaml 中 trident 相关配置：

【可选】用于部署 ESXi 环境中的采集器虚拟机，以列表存在，如有多个就填写多个列表，以下为 2 个列表参考

```
tridentConfigmap:
  ## 主机名
  - name: __FIXME_TRIDENT_HOSTNAME1__
    ## Controller IP
    ip: __FIXME_TRIDENT1_IP__
    ## 网卡名列表
    tapInterfaces:
      - eth1
      - eth2
  - name: __FIXME_TRIDENT_HOSTNAME2__
    ip: __FIXME_TRIDENT2_IP__
    tapInterfaces:
      - eth1
      - eth2
```


注意： - 目前不支持 vSphere 环境下业务虚拟机运行 K8s 容器的虚拟网络链路追踪展示 - 当业务虚拟机中运行容器且安装了容器类型采集器时，请勿将该虚拟机的流量镜像至 ESXi 类型采集器

部署 ESXi Trident 流程

- VMware 虚拟机扩容为 K8s node
- node 打上 trident=enable 标签
- 修改 vaues-custom.yaml 修改 ESXi 采集器相关配置
- k8s master 节点上执行安装 DeepFlow
- 如已安装 DeepFlow，新增 ESXi trident 部署，则使用升级命令
`/usr/local/deepflow/bin/deepflow-deploy -u`
- `kubect1 get pods -n deepflow -o wide | grep trident` 检查 trident 是否运行在 VMware 虚拟机节点上

容器

依赖检查

主要功能采集 K8s 所有容器的流量，目前容器部署只支持 K8s 环境。

采集配置

trident.yaml 配置文件： - 推荐：将 `vtap-group-id-request` 设置为系统-采集器-组页面的某个 ID 字段值，使得采集器被发现时可自动加入特定的采集器组

系统-采集器-配置页面： - 采集网口配置为采集流量的网口的正则表达式，请添加^强制匹配开头，例如：`^(eth|veth).*-eth.*`：当 K8s 的 CNI 跨节点通信使用隧道时，需要配置采集网口包含 K8s 节点的出网口（例如物理机的物理网口），以展现虚拟网络链路追踪能力 - `veth.*`：POD 的虚拟网卡 - `lo`：当两个 HostNetwork 的 POD 互访时时走 `lo` 口 - 注意：采集网口配置必须遵循只看两端不看中间的原则，除 `lo` 接口上的本机流量以外，仅采集容器 POD、KVM 宿主机或物理交换机进出容器节点的流量，请勿配置除此之外的其他网口，否则会导致采集压力上升，并导致页面显示数据错误。 - 隧道解封装配置为正确的 SDN 隧道协议，以展示虚拟网络链路追踪能力 - 当 K8s 的 CNI 跨节点通信使用隧道时，根据隧道类型选择 `VXLAN` 或 `IPIP` - 目前不支持其他隧道封装协议

权限要求

必须开放的权限

- HostNetwork

决定容器是否共享宿主的 `net` 命名空间，设置为 `true` 后容器不会使用单独的 `net` 命名空间。Trident 采集主机命名空间中 `vEth Pair` 接口、`Node` 接口流量需要该权限。

- API 读取权限

trident 对接 kubernetes 平台还需要以下资源的 get/list/watch 权限

- nodes
- namespaces
- kube-system 命名空间 configmaps
- services
- pods
- replicationcontrollers
- deployments
- statefulsets
- daemonsets
- replicaset
- ingresses (kubernetes) | routes (openshift)

一个示例配置如下：

```
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: deepflow-trident-role
rules:
- apiGroups: [""]
  resources:
    - nodes
    - namespaces
    - configmaps
    - services
    - pods
    - replicationcontrollers
  verbs: ["get", "list", "watch"]
- apiGroups: ["apps"]
  resources:
    - daemonsets
    - deployments
    - replicaset
    - statefulsets
  verbs: ["get", "list", "watch"]
- apiGroups: ["networking.k8s.io"]
  resources: ["ingresses"]
  verbs: ["get", "list", "watch"]
- apiGroups: ["route.openshift.io"]
  resources: ["routes"]
  verbs: ["get", "list", "watch"]
```


建议开放的权限

容器默认以非特权模式运行，无法访问任何设备。使用`--privileged`开关可以将容器运行在特权模式，此时容器拥有的权限基本和在容器外的宿主上一致。非特权模式默认给予了一部分权限，可以通过`--cap-drop`去掉；相对的也可以通过`--cap-add`增加权限，参考[此链接](#)。

Trident 在容器中运行时，建议开放以下四个权限：

- **SYS_ADMIN**

这个权限最初设计是用于系统管理，但随着 Linux 发展变成了比较繁杂的一个权限。具体的权限列表见[此链接](#)。特权模式和该权限的区别之一是，特权模式会将`/dev/`和`/sys`以读写模式挂载，而 **SYS_ADMIN** 会挂载为只读，前者对系统中的设备有完整的访问权限。Trident 需要该权限是为了在 Kubernetes 环境中调用 `setns` 切换命名空间，以获取 `net` 命名空间中的接口 MAC 和 IP 地址。

- **SYS_PTRACE**

该权限用于在 `pod` 间共享命名空间。由于 `trident` 在获取接口信息时需要访问其他 `pod` 的网络命名空间，所以该权限需要开启。

- **NET_ADMIN**

该权限用于配置 `pod` 网络，如接口配置，开启混杂模式。

- **HostPID**

决定容器是否共享宿主的 `PID` 命名空间，设置为 `true` 后容器内可以看到宿主上所有的进程。Trident 需要该权限是在使用 `setns` 前需要通过 `PID` 拿到相应的命名空间。

建议： Trident 也可以在非 **SYS_ADMIN**、**SYS_PTRACE** 或 **HostPID** 模式下运行，此时平台信息采集会采用从接口流量获取的方式，时效性和准确率不如特权模式，且子网信息无法推断。

警告： 在极少数场景下发现即使赋予了 **SYS_ADMIN** 和 **SYS_PTRACE** 权限，Trident 依然无法访问某些 `pod` 的网络命名空间，此时 Trident 日志中会打印如下信息：

```
WARN: kubernetes poller privileges: set_ns=true readlink_ns=false
WARN: kubernetes poller in passive mode
```

在启动容器采集器前，在容器节点上使用以下命令确认 **SYS_ADMIN** 和 **SYS_PTRACE** 权限是否足够

```
docker run --cap-add SYS_ADMIN --cap-add SYS_PTRACE --cap-add NET_ADMIN --pid host --network host deepflow-trident:{version} trident -check-privileges
```

启动容器采集器后也可以在 k8s master 上使用如下命令进行检查

```
kubectl -n yunshan-deepflow exec -it {trident-pod} -- trident --check-privileges
```

若权限不足（表现为在资源-容器资源-容器 POD 列表中部分 POD 无法显示 MAC 地址信息），建议改为使用 `privileged` 权限，对 `daemonset` 的配置需手动修改如下（可参考 `daemonset` 配置文件中的注释）：

```
securityContext:
  privileged: true
```

OpenShift 部署注意事项

- 配置 `ClusterRole` 时需要注意 OpenShift 中没有 `ingress` 资源，与之对应的是 `route` 资源

```
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: deepflow-trident-role
rules:
- apiGroups: [""]
  resources:
    - nodes
    - namespaces
    - configmaps
    - services
    - pods
    - replicationcontrollers
  verbs: ["get", "list", "watch"]
- apiGroups: ["apps"]
  resources:
    - daemonsets
    - deployments
    - replicaset
    - statefulsets
  verbs: ["get", "list", "watch"]
- apiGroups: ["route.openshift.io"]
  resources: ["routes"]
  verbs: ["get", "list", "watch"]
```

- 需要使用以下命令增加 SCC

```
oc adm policy add-scc-to-user privileged -n yunshan-deepflow -z default
```

安装 Trident

- 获取镜像，执行如下命令，安装 trident_docker RPM 包：

```
yum --disablerepo=* --enablerepo=deepflow -y install trident_docker
```

- 修改 trident-cm.yaml 配置文件：

```
# 安装完成后，会在/tmp/生成trident目录，直接进入该目录
cd /tmp/trident
```

编辑 trident-cm.yaml 文件：

```
controller_ips:
  - 127.0.0.1      # 修改为主控制器的IP
  - 127.0.0.2      # 修改为备控制器的IP
## 注意：如下两行仅在mac-vlan模式下需要修改
# tap-mode: 1      # 指定为mirror模式
# src-interfaces: [ethX] # ethx为mac-vlan的物理口
kubernetes-cluster-id: xxx # 修改从“资源-资源池-云平台”中查看到的ID，
                             形如“cl-xxxxxxxxxx”
ingress-flavour: kubernetes/openshift # 修改为对应的容器平台
```

- 创建 namespace yunshan-deepflow，供 Trident 的 DaemonSet 使用：

```
kubectl create namespace yunshan-deepflow
```

- 创建 K8S ConfigMap 资源（用于管理容器内部的配置文件，可以理解为 Linux 中/etc 目录）：

```
kubectl apply -f trident-cm.yaml
```

- 上传 Trident image

- K8S 环境内，没有私有仓库，操作步骤如下：

- 将 deepflow-trident-*.tar 文件 copy 到 k8s 所有 node 节点：

```
scp deepflow-trident-*.tar $node:/root
```

- 导入 trident 镜像

```
# 登录到每个node节点，并将trident镜像导入到本地
ssh $node
docker load -i /root/deepflow-trident-*.tar
```

- 为 trident 镜像打上标签，以 deepflow-trident-1.0-1.tar 为例：

```
docker tag deepflow-trident:1.0-1 deepflow-trident:latest
```

- 修改 trident-ds.yaml 文件

```
imagePullPolicy: Never           #将此参数改为Never，表示
                                  使用本地镜像，默认为Never
image: deepflow-trident:latest  #修改为deepflow-trident:
                                  latest
```

- k8s 环境内有私有仓库，部署步骤如下，该步骤可由客户操作，涉及到私有仓库的账号密码

- 将 trident 镜像导入到本地

```
docker load -i deepflow-trident-*.tar
```

- 为 trident 镜像打上标签，以 deepflow-trident-1.0-1.tar 为例：

```
docker tag deepflow-trident:1.0-1 $registry_ip:$registry_port/deepflow-trident:latest
# registry_ip: 私有仓库的 IP 地址，也可以是主机名，域名，需要从客户那边获取信息，如果是域名，无需指定 registry_port
# registry_port: 私有仓库绑定的端口号，需要从客户那边获取
```

- 将打上标签的镜像上传到私有仓库

```
docker push $registry_ip:$registry_port/deepflow-trident:latest
```

- 修改 trident-ds.yaml 文件

```
imagePullPolicy: Always           # 该参数修改为Always，表示永远更新远端的 latest 镜像，当远端没有对应标签的 image 时，会启动失败。
image: 10.0.0.12:5001/deepflow-trident:latest # 修改为打标签后的镜像为$registry_ip:$registry_port/deepflow-trident:latest
```

- 启动容器

```
kubectl apply -f trident-ds.yaml
# 查看 trident 容器是否启动,$tridentname 修改 containers.name 中设置的名称即可
kubectl get pod $tridentname -n yunshan-deepflow # 查看 STATUS 状态为 running 状态，即为启动，启动之后，可正常采集流量测试
```

调试方法

在 DeepFlow 节点上安装 kubectl

kubectl 安装（561 R2 开始,kubectl 的安装包会集成到 ISO 中）：

```
yum install -y kubectl --disablerepo=* --enablerepo=deepflow
```

新建/root/.kube/config: vim /root/.kube/config, 并录入如下内容（注意修改三行注释下的内容）：

```
apiVersion: v1
clusters:
- cluster:
  # 将 /etc/cloud-agent.yaml 文件中 k8s 配置部分的 apiserver_url 内容填入下面的 server 字段中
  server: https://140.143.129.9:6443
  name: kubernetes
contexts:
- context:
  cluster: kubernetes
  user: kubernetes-admin
  name: kubernetes-admin@kubernetes
current-context: kubernetes-admin@kubernetes
kind: Config
preferences: {}
users:
- name: kubernetes-admin
  user:
  # 将如下命令的输出填入下面的 client-certificate-data 字段中, 注意命令中的 crt 文件会在 k8s 平台对接时拿到
  # cat apiserver-kubelet-client.crt | base64
  client-certificate-data:
  # 将如下命令的输出填入下面的 client-key-data 字段中, 注意命令中的 key 文件会在 k8s 平台对接时拿到
  # cat apiserver-kubelet-client.key | base64
  client-key-data:
```

常用命令

部署 Trident 后, 可使用如下命令查看容器节点上的信息:

```
#### 查询 Trident 所在命名空间（假设为 deepflow）下的 POD 信息
kubectl get pods -n deepflow -o wide
```

```
#### 挑选一个 POD 进入其 shell
kubectl exec -it trident-45kmv -n deepflow sh
```

```
#### 从 POD 的 shell 进入容器节点的 shell
nsenter -t 1 -m -u
```

使用 kubectl 查看 Trident POD 是否正常（因为 config 文件中未加 CA 根证书需要跳过证书确认）：

```
kubect1 get pod -n yunshan-deepflow --insecure-skip-tls-verify=true -o wide
```

临时停用/启用采集器 Daemonset:

将Daemonset 禁用

```
kubect1 -n yunshan-deepflow patch daemonset trident -p '{"spec": {"template": {"spec": {"nodeSelector": {"non-existing": "true"}}}}}'
```

启用Daemonset

```
kubect1 -n yunshan-deepflow patch daemonset trident --type json -p='[{"op": "remove", "path": "/spec/template/spec/nodeSelector/non-existing"}]
```

专属服务器

安装专属采集器

- 部署时修改 values-custom.yaml 文件中 dedicatedTrident 相关内容
- 对节点打上 dedicated-trident label，安装时将随 DeepFlow 一起安装专属采集器
- 如已安装 DeepFlow，新增 trident 部署，则修改完 values-custom.yaml 及打完 label 后使用升级命令 /usr/local/deepflow/bin/deepflow-deploy -u

采集配置

资源-其他资源-采集点： - 在页面上录入采集点信息

trident.yaml 配置文件： - 配置 mirror-traffic-pcp 为镜像流量 VLAN 头部的 PCP 值，只有匹配该值的流量才会根据 VLAN 值标记对应的采集点 - 当镜像流量没有统一的 PCP 值时，配置 default-tap-type 为页面录入采集点的数据标记值，表示将镜像流量全部归入该采集点 - 当采集器接收交换机的 ERSPAN 镜像流量时，需要显示指定 decap-erspan: true - 推荐：使用 b1d 提前标记 ERSPAN 隧道头部长度，提升 trident 解封隧道的性能，参考本页下一小节 - 推荐：当镜像虚拟网络（tap-type=3）的流量时，若确认流量没有重复，可通过设置 analyzer-dedup-disabled: true 提升性能 - 推荐：将 vtap-group-id-request 设置为系统-采集器-组页面的某个 ID 字段值，使得采集器被发现时可自动加入特定的采集器组

系统-采集器-配置页面： - 隧道解封装配置为正确的 SDN 隧道协议，以展示虚拟网络链路追踪能力 - 对于腾讯专有云（TCE）场景，当镜像网关区流量且作为虚拟网络采集点时，需要配置为 TCE GRE

警告： 流量镜像注意事项：由于 QinQ 包无法让采集器网卡实现 RSS 均摊，导致中断集中在一个 CPU 上，极易发生丢包。流量镜像时请使用 VLAN 改写的机制，确保镜像流量最多只有一层 VLAN。

bild 部署和配置[可选]

当大流量过大或出现丢包时需要安装 *bild*，目前 *bild* 只支持 4.19 版本内核 #####
salt 环境下部署

- 部署 *deepflow* 时，*bild* 默认处于 disable 状态。
- 如需 enable *bild*，需修改 `/usr/local/deepflow/saltstack/pillar/bild.sls` 中的 *bild* 配置

```
bild:
  dedicated-trident:
    enable: False
    dst_interface_name_prefix: tap
    dst_interface_count: 8
  trident:
    enable: False
    dst_interface_name_prefix: tap
    dst_interface_count: 1
```

- *bild* 运行在 *trident* 和 *dedicated-trident* 节点
- enable: False
 - enable/disable *bild* 功能，默认 False，如果启用使用 True
- *dst_tap_interface_name_prefix*: tap
 - 创建虚拟接口的接口名前缀，默认 tap
- *dst_tap_interface_count* 8/1
 - 创建的虚拟接口数量，*dedicated-trident* 节点默认 8 个，*trident* 节点默认 1 个。
- 然后将 *bild* 部署到相应的节点

- 部署到 *dedicated-trident* 节点：

```
# 配置文件更新
salt-run state.orch dedicated_trident_config
# 关闭trident、bild
salt-run state.orch dedicated_trident_stop
# 启动trident、bild
salt-run state.orch dedicated_trident_start
```

- 部署到 *trident* 节点：

```
# 配置文件更新
salt-run state.orch trident_config
# 关闭trident、bild
salt-run state.orch trident_stop
# 启动trident、bild
salt-run state.orch trident_start
```

容器环境下部署

- 获取 bild 和 iproute 安装包
 - 从 ISO 中的 extra-artifacts/df-bild-package 获取。
- 安装

```
yum --disablerepo=* localinstall -y ./{iproute,bild}/*.rpm
```

说明: bild 对系统包 glibc 有一定的要求, 需要 $\geq 2.17-260$ 如果版本过低需要通过 yum update glibc 进行更新。

- 创建虚接口
 - 在文件 /sbin/ifup-local 追加下面, 添加权限并执行 chmod +x /sbin/ifup-local; sh /sbin/ifup-local

```
for i in {0..7}
do
    /sbin/ip link add tap-bild-$i type dummy
    /sbin/ip link set tap-bild-$i promisc on
    /sbin/ip link set dev tap-bild-$i up
done
```

- 修改配置文件 /etc/bild.conf(下面举例)

```
ebpf:
- src_tap_interfaces: em1
  src_direction: 0
  dst_tap_interfaces: tap-bild-0,tap-bild-1,tap-bild-2,tap-bild-3,
tap-bild-4,tap-bild-5,tap-bild-6,tap-bild-7
  dst_direction: 0
```

- 启动 bild

```
systemctl enable bild.service
systemctl start bild.service
```

- 修改 trident 采集接口的配置 (修改成上面建立的 8 个虚拟接口) 并重启 trident。

配置网卡优化功能

- 根据实际需求可以对网卡做对称哈希, XDP CPU 重定向等优化。根据实际情况可以对某个节点单独修改 /etc/bild.conf 文件。

```
nic_optimize:
- interface: p1p2
  rss_channel_count: 1
  hi_cpu_list: 2
```



```
xdp_cpu_redirect: true
xdp_cpu_redirect_list: 5,6,7,8,9,10
```

以上配置是样例，具体配置含义参照 NIC 优化配置章节。

- 某个节点添加配置后，重启 bild。systemctl restart bild

配置文件/etc/bild.conf 配置选项说明

配置接口均衡转发

ebpf:

```
- src_tap_interfaces: lo
  src_direction: 0
  dst_tap_interfaces: lo
  dst_direction: 0
```

- src_tap_interfaces
 - 源接口
 - 最大支持 32 个接口
 - 接口直接用,分割且不能有空格、tab 等空白字符
- dst_tap_interfaces
 - 目的接口
 - 最大支持 128 个接口，因为 trident 目前最多支持 16 个接口，所以实际设置接口的数量不要超过 16。
 - 接口直接用,分割且不能有空格、tab 等空白字符
- src_direction
 - 源接口流量方向
 - 0: ingress, 1: egress
- dst_direction
 - 目的接口流量方向
 - 0: ingress, 1: egress

NIC 优化配置

nic_optimize:

```
- interface: p1p2
  rss_channel_count: 1
  hi_cpu_list: 2
  xdp_cpu_redirect: true
  xdp_cpu_redirect_list: 5,6,7,8,9,10
```

- 如果不进行网卡优化可注释掉上面配置选项，默认：不启用网卡优化，处于注释状态。
- interface
 - 要优化配置接口的名称，如：interface: p1p2

- 这个接口是 `src_tap_interfaces` 中的一个，不能写多个如： `interface: p1p2,p2p1`，如果要进行多个接口做优化可以如下配置：

```
nic_optimize:
- interface: p1p2
  .....
- interface: p1p1
  .....
```

- `rss_channel_count`
 - RSS 硬件队列数量，数据流在物理网卡上做哈希之后，均摊到多少个队列上并在此队列上发出中断，最大可配置 16 且不要超过逻辑 CPU 核数，当启用 XDP CPU Redirect 时配置为 1
- `hi_cpu_list`
 - 中断和 CPU 绑定，值：CPU-ID/CPU-ID 列表
 - 处理硬件中断的 CPU 列表或单个 CPU，数量最好等于 `rss_channel_count`，但流量较小时一个 CPU 可处理多个硬件中断队列，当启用 XDP CPU Redirect 时只需指定 1 个 CPU 即可。
- `xdp_cpu_redirect`
 - 是否开启 XDP CPUMAP redirect 功能。值：true/false
- `xdp_cpu_redirect_list`
 - XDP CPU Redirect 后处理包的 CPU 列表

注意： 注：修改配置文件后，需重启 `bird` 才能生效

业务交换机配置

建议： 当需要镜像一个业务交换机所有流量时，推荐对所有接口的出方向进行镜像：
- 避免将多个接口的流量全部镜像到一个镜像口中，采集器收到的流量会有重复 -
避免只镜像所有接口的入方向流量，镜像出方向更有利于判断丢包位置

业务交换机镜像配置

盛科

```
monitor session 1 destination interface eth-0-48
monitor session 1 source interface eth-0-1 both
```

华为

```
observe-port 1 interface GigabitEthernet0/0/1
interface GigabitEthernet0/0/14
  port-mirroring to observe-port 1 inbound
  port-mirroring to observe-port 1 outbound
```

Arista

```
monitor session name source Ethernet 12
monitor session name destination Ethernet 1
```

业务交换机 sFlow 配置

以华为交换机为例： - 创建 sFlow Agent 并配置 sFlow Agent 的 IP 地址 - 配置 sFlow 报文的源 IP 地址 - 创建 sFlow Collector 并配置 sFlow Collector 相关参数 - 配置接口 sFlow 采样的 Collector ID, 并指定接口入方向流量采样

```
sflow agent ip 10.20.30.40
sflow source ip 10.20.30.50
sflow collector 1 ip 10.20.30.60 udp-port 6343 length 1400
interface 10GE1/0/1
sflow sampling collector 1 inbound
```

业务交换机 NetFlow/NetStream 配置

以华为交换机为例： - 配置流统计信息输出报文的源地址 - 配置流统计信息输出报文的地址和目的 UDP 端口号 - 配置流的活跃老化时间，推荐设置支持的最小值 - 使能入接口的流量统计功能 - 配置接口的随机报文采样的间隔, 并指定对接口入方向的流量进行采样

```
netstream export ip source 10.20.30.40
netstream export ip host 10.20.30.50 55555
netstream timeout ip active 1
netstream export ip template timeout-rate 1
interface 10GE1/0/1
netstream inbound ip
netstream sampler random-packets 1 inbound
```

TAP 交换机配置

本节描述以普通交换机当做 TAP 交换机时的配置。

H3C 交换机

警告： 下述参考配置默认使用 QinQ 的方式为镜像流量打标记，这种方式已从 2020 年 8 月开始废弃。原因是 QinQ 包无法让采集器网卡实现 RSS 均摊，使得中断集中在一个 CPU 上导致丢包。新部署的系统，请使用 VLAN 改写的机制，避免生成 QinQ 流量。

基础配置

空配置的 H3C 交换机只能够通过串口连接，因此需要 RS232 接口连接到交换机的 Console 接口，并设定波特率为 9600。

```
#### 进入系统视图
system-view
```

```
#### 配置 MGE 接口 IP 地址
```

```
#### 注：MGE 接口的编号可能随 IRF 配置的状态而改变
```

```
interface M-GigabitEthernet0/0/0
  ip address 172.16.1.130 255.255.0.0
  quit
```

```
#### 启用 SSH Server
ssh server enable
```

```
#### 添加 SSH 用户
local-user USER class manage
  service-type ssh
  authorization-attribute user-role network-admin
  password simple PASSWORD
  quit
```

```
#### 设定 vty 终端权限
user-interface vty 0 4
  authentication-mode scheme
  user-role network-admin
  quit
```

接收业务交换机镜像流量

```
interface Bridge-Aggregation 2
  quit
interface range Ten-GigabitEthernet 1/0/41:1 to Ten-GigabitEthernet 1/0/41:4
  qos priority 7
  port link-aggregation group 2
  quit
interface Bridge-Aggregation 2
  # 注意在物理口加入聚合组以后，需要刷新聚合口配置
  undo stp enable
  qinq enable
  port access vlan 257
  mac-address mac-learning disable
  quit
```

分发业务交换机镜像流量

```
interface Bridge-Aggregation 1
  quit
interface range Ten-GigabitEthernet1/0/17 to Ten-GigabitEthernet1/0/18
  port link-aggregation group 1
  quit
interface Bridge-Aggregation 1
  # 注意在物理口加入聚合组以后，需要刷新聚合口配置
  port link-type trunk
  port trunk permit vlan 257
  link-aggregation load-sharing mode source-ip destination-ip
  mac-address mac-learning disable
  quit
```

VXLAN 配置

可选，仅用于网包分发功能的隧道解封装设备。

设置交换机工作在 vxlan 模式，并配置 vxlan 隧道模式为二层转发模式

```
switch-mode 1
undo vxlan ip-forwarding
```

开启 l2vpn 功能

```
l2vpn enable
```

配置控制平面 VLAN Interface

```
interface vlan-interface 2
  ip address 172.20.1.53 255.255.0.0
```

创建模式为 vxlan 隧道的 tunnel 接口

```
interface tunnel 1
  source 172.20.1.53
  destination 172.20.1.54
```

创建 VSI

```
vsi vpna
  vxlan 10000
  tunnel 1
```

连接第三方设备的物理接口

```
interface x/x/x
  service-instance 10000
  encapsulation s-vid xx
  xconnect vsi vpna
```

关闭远端 MAC 地址自动学习

```
vxlan tunnel mac-learning disable
```

其他配置

如果需要将 40GE 光口拆分为 4*10GE，请参考如下命令

```
interface interface FortyGigE1/0/X
  using tengige
  Y
  quit
```

Huawei 交换机

警告： 下述参考配置默认使用 QinQ 的方式为镜像流量打标记，这种方式已从 2020 年 8 月开始废弃。原因是 QinQ 包无法让采集器网卡实现 RSS 均摊，使得中断集中在一个 CPU 上导致丢包。新部署的系统，请使用 VLAN 改写的机制，避免生成 QinQ 流量。

基础配置

按需更新交换机镜像，以 CE 交换机为例：

在普通模式下可使用 `dir` 命令查看文件和 `flash` 的剩余空间

`dir`

若空间不足，在普通模式下可使用 `delete` 删除文件并 `reset` 清理回收站

`delete flash:/CE5810EI-V100R003C00SPC600.cc`

`reset recycle-bin`

在管理模式下可使用 `scp` 从远端服务器上拷贝镜像文件和 `patch` 文件到 `flash`

`scp root@192.168.18.2:/home/xiangyang/CE5810EI-V200R001C00SPC700.cc flash:/`

`scp root@192.168.18.2:/home/xiangyang/CE6851HI-V200R001SPH008.PAT flash:/`

在普通模式下设置启动镜像并重启

`startup system-software flash:/CE5810EI-V200R001C00SPC700.cc`

`startup patch CE6851HI-V200R001SPH008.PAT all`

在普通模式下检查启动项设置是否正确

`display startup`

在普通模式下重启交换机

`reboot`

Hotpatch 单独更新方法

在管理模式下可使用 `scp` 从远端服务器上拷贝 Hotpatch 文件到 `flash`

`scp root@192.168.18.2:/home/xiangyang/CE6851HI-V200R001SPH008.PAT flash:/`

在普通模式下加载并运行 Hotpatch

`patch load CE6851HI-V200R001SPH008.PAT all run`

在普通模式下检查 patch 信息

`display patch-information`

交换机管理

`interface MEth0/0/1`

`ip address 172.17.1.100 255.255.0.0`

配置 SNMP

`snmp-agent`

`snmp-agent community read zabbix`

`snmp-agent sys-info version v2c`

用户及权限，此部分仅为举例，不同型号的具体配置可能有差异

`ssh authentication-type default password`

`aaa`

`local-user admin password cipher the-login-password`

`local-user admin privilege level 15`

`local-user admin service-type ssh`

`# 关闭密码安全性检查`

```
undo local-user policy security-enhance
quit

stelnet server enable
ssh user admin authentication-type password
ssh user admin service-type stelnet
user-interface vty 0 4
    authentication-mode aaa
    protocol inbound ssh

#### 关闭 STP
stp disable

#### 打开 LLDP
lldp enable
lldp management-address 172.17.1.100 # Management IP

#### 使能 Netconf API
netconf
    protocol inbound ssh port 830
snetconf server enable
ssh user admin service-type all

#### 使能 VLAN: 打开控制、数据平面用到的所有 VLAN
vlan batch 100 256 to 2047

#### 确保所有端口已打开 jumboframe, 即 Maximum Frame Length 大于 9000
display interface | include The Maximum Frame Length is
#### 若不大于 9000, 需要对所有相应的接口执行 jumboframe 打开的命令
interface 10GE 1/0/1
    jumboframe enable

#### 配置统计数据计算间隔
set flow-stat interval 60

#### 去使能 SSH 服务器上的客户端 IP 地址锁定功能
ssh server ip-block disable
```

接收业务交换机镜像流量

```
interface Eth-Trunk2
    port link-type dot1q-tunnel
    port default vlan 257
    port priority 7
    mac-address learning disable
    traffic-policy remark-isp inbound
quit
interface GE1/0/1
```

```
eth-trunk 2
quit
```

分发业务交换机镜像流量

```
load-balance profile deepflow-npb
  ip src-ip dst-ip protocol
  l2 eth-type
interface Eth-Trunk1
  # 注意：华为交换机 resilient 功能存在 Bug，暂停使用
  # 仅 CE8860-EI/CE7850-EI/CE6850U-HI/CE6851-HI/CE6850-HI 支持 resilient
  参数，
  # 其他型号交换机去掉 resilient 关键字即可。
  #load-balance enhanced resilient profile deepflow-npb
  load-balance enhanced profile deepflow-npb
  port link-type trunk
  port trunk allow-pass vlan 257
  mac-address learning disable
  description deepflow-tool-dfi-0
interface 10GE1/0/1
  eth-trunk 1
interface 10GE1/0/2
  eth-trunk 1
```

VXLAN 配置

可选，仅用于网包分发功能的隧道解封装设备。举例：采集器 Trident 源 IP 为 172.16.1.1，VNI 为 1024，在 172.20.1.53 解封装。

```
####
vlan batch 2 1024
####
interface Vlanif 2
  ip address 172.20.1.53 16
  quit
####
bridge-domain 1024
  l2 binding vlan 1024
  vxlan vni 1024
  quit
####
interface Nve 1
  mac-address learning disable
  source 172.20.1.53
  vni 1024 head-end peer-list 172.16.1.1
  quit
####
```


ECMP 配置

假设有两台采集器，其控制 ip 分别为 172.16.0.2 和 172.16.0.3，虚 IP 为 172.18.0.1 流量分发交换机的 ECMP 配置方式如下（假设采集流量从 GE1/0/15 进入）：

```
system-view

interface Vlanif 2
 ip address 172.20.1.1 16
 quit

ip route-static 172.18.0.1 255.255.255.255 172.16.0.2
ip route-static 172.18.0.1 255.255.255.255 172.16.0.3

interface range 10GE 1/0/2 to 10GE 1/0/3
 # 提高控制平面优先级
 port priority 7
 quit

commit
```

其他配置

如果需要将 40GE 光口拆分为 4x10GE，请参考如下命令

```
#### 进入系统视图
system-view
port split dimension interface 40GE 1/0/8 split-type 4*10GE
commit
#### 进入普通视图
quit
save
#### 重启设备生效
reset slot 1
```

Centec 交换机

警告： 下述参考配置默认使用 QinQ 的方式为镜像流量打标记，这种方式已从 2020 年 8 月开始废弃。原因是 QinQ 包无法让采集器网卡实现 RSS 均摊，使得中断集中在一个 CPU 上导致丢包。新部署的系统，请使用 VLAN 改写的机制，避免生成 QinQ 流量。

基础配置

####. 按需更换或升级交换机镜像

从远端 FTP 服务器上拷贝镜像，可使用 vsftpd 搭建 FTP 服务器，注意防火墙放开相关端口

```
ftp mgmt-if 172.16.0.85
```

```
ftp> get centecOS-e580-v6.2.22.fcs.r.bin
#### 将镜像移动到 flash:/boot/下
copy flash:/centecOS-e580-v6.2.22.fcs.r.bin flash:/boot/
delete flash:/centecOS-e580-v6.2.22.fcs.r.bin
#### 设置启动镜像并重启
boot system flash:/centecOS-e580-v6.2.22.fcs.r.bin
reboot

#### 交换机管理
hostname SW-CRA-Cen35-POD1-PB19-40
management ip address 172.17.1.40/16
service rpc-api enable
#### 配置 SNMP 用于 Zabbix 监控
snmp-server enable
snmp-server community zabbix read-only

#### 使能 errdisable 自动恢复，避免 errdisable 被触发后端口一直处于 down 状态
errdisable recovery interval 30
errdisable recovery reason link-monitor-failure
errdisable recovery reason link-flap
errdisable recovery reason fdb-loop

#### 打开全局 LLDP 和所有端口 LLDP
lldp enable
lldp management ip 172.17.1.40
interface eth-0-1
    lldp enable txrx

#### 关闭 STP
no spanning-tree enable

#### 用户及权限
ip ssh server authentication-type password
username admin privilege 4 password ssh-password
line vty 0 7
    # 如果发现没有下面的配置，需要添加配置、保存并 reboot
    privilege level 4

#### 按需使能 VLAN
vlan database
    vlan 100
    vlan 256-2047

#### 确保所有端口已打开 jumboframe，即 Maximum Frame Length 大于 9000
show interface | include The Maximum Frame Size is
#### 若不大于 9000，需要对所有相应的接口执行 jumboframe 打开的命令
interface eth-0-1
    jumboframe enable
```

设置负载均衡字段为 IP+协议

```
port-channel load-balance hash-field-select ipsa ipda ip-protocol
```

接收业务交换机镜像流量

```
interface eth-0-51
    static-channel-group 1
interface eth-0-52
    static-channel-group 1
interface agg1
    description tap-isp-1
    switchport mode dot1q-tunnel
    switchport dot1q-tunnel native vlan 257
    switchport dot1q-tunnel allowed vlan add 257
    switchport dot1q-tunnel allowed vlan remove 1
    mac learning disable
```

E580 需要额外配置聚合组的每个成员接口

```
interface eth-0-51
    service-policy type qos input pmap-df-cos
interface eth-0-52
    service-policy type qos input pmap-df-cos
```

E350 需要额外配置聚合组

```
interface agg1
    trust port
    cos 7
```

分发业务交换机镜像流量

```
interface eth-0-49
    static-channel-group 3
interface eth-0-50
    static-channel-group 3
interface agg3
    description dfi-0
    switchport mode trunk
    switchport trunk allowed vlan add 257
    mac learning disable
    # 上面一行的命令在 E350 中配置方法如下，仅供参考
    # switchport trunk allowed vlan add 257
```

VXLAN 配置

可选，仅用于网包分发功能的隧道解封装设备。

配置控制平面 VLAN Interface

```
interface Vlanif2
    ip address 172.20.1.53 255.255.0.0
overlay
```

```
source 172.20.1.153
#### VLAN 1024 至 2047 关闭 MAC 学习，由于命令较多，测试环境可按需关闭
vlan database
    vlan 1024 mac learning disable
#### 控制平面上行链路：和引流器相连的链路
interface eth-0-1
    overlay uplink enable
```

其他配置

COS 相关配置（针对 CentecOS Software Version 6 及以上，即 E580）：

```
mac access-list acl-df-cos
    permit src-mac any dest-mac any
class type qos cmap-df-cos
    match access-group acl-df-cos
policy-map type qos pmap-df-cos
    class type qos cmap-df-cos
        set cos 7
```

Workload

采集配置

trident.yaml 配置文件： - 推荐：将 vtap-group-id-request 设置为系统-采集器-组页面的某个 ID 字段值，使得采集器被发现时可自动加入特定的采集器组 - IBM S390 CPU 架构时需要使用 libpcap 抓包，并向研发获取定制 RPM 包： - 开启 libpcap: switch-to-pcap: true - 设置抓包接口：src-interfaces: [ethX]

系统-采集器-配置页面： - 采集网口：配置为采集流量的网口的正则表达式，请添加^强制匹配开头 - 注意：Linux 下建议加上 lo 口以采集本机两个进程之间的通信 - 裸 UDP 最大 MTU：目前发现在部分公有云中发送超过 1400 字节的 UDP 包时，末尾内容可能会被修改，推荐在公有云场景下将该值配置为 1400 - 解封装隧道类型：对于微软 ACS 中 MUX 虚拟机的采集器需要根据流量隧道类型配置该字段

在 Linux Workload 中运行 Trident

依赖检查

每个 Guest VM 内部运行一个 trident。注意检查 trident 与控制器、数据节点控制平面的连通性

安装 Trident

和 KVM 下的安装方式相同，由用户自行安装。测试环境下可使用 mt.sh 部署。

在 Windows Workload 中运行 Trident

依赖检查

检查 VM 的 C:\Windows\System32 目录下是否有 wpcap.dll，否则需要安装 wpcap.dll，推荐使用 Win10Pcap。检查宿主机是否安装 npf 服务，使用命令 `sc query npf`

安装 trident

- 从发布 ISO 的 windows 目录将 trident-setup.exe 拷贝到云服务器中
- 双击 trident-setup.exe 进行安装，注意安装路径不能包含空格，在 Completing trident Setup 页面，先不选择 Run trident，修改完配置文件再启动 trident
- 到安装目录修改 trident-windows.yaml 配置文件，修改控制器 IP
- 左下角搜索框输入 service，打开服务
- 找到 DeepFlow Trident 服务，双击
- 点击启动，启动 trident 进程
- 到恢复标签页，将第一次失败、第二次失败、后续失败均修改为重新启动服务
- 点击应用、确定退出配置服务界面
- 若重新安装或者升级 trident，应先停止 DeepFlow Trident 服务，再安装 trident-setup.exe，安装完成后启动服务

隧道解封装组件

简介

本指南为 DeepFlow 采集器-Vxlan 隧道卸载模式（river/rosen）的部署和配置方法。

当用户的第三方工具不能解析来自 Trident 分发过来的 VxLAN 流量时，river 和 rosen 可用于隧道解封装。二者的差别在于：- river 可安装于任意 Linux 操作环境下，无任何运行依赖，性能极限为 2Gbps / 200Kpps - rosen 需要运行于独立的物理或虚拟服务器中，且服务器需要安装 DeepFlow 提供的 Linux 内核，性能极限为 10Gbps / 10Mpps

注意：- 不支持运行于 DPDK 环境中 - 不支持 VXLAN 以外的协议解封装

部署安装 - River

将控制器中的 river*.rpm 复制到分发节点上：

```
scp deepflow-controller:/usr/local/deepflow/yum/7/river*.rpm /tmp/
```

安装 river：

```
yum --disablerepo=* localinstall -y /tmp/river*.rpm
```

添加 tap0 虚拟口用于发送解隧道后的流量：

```
ip link add tap0 type dummy
ip link set tap0 up
```

警告： 警告：请勿将解封装流量发送到物理接口，有可能会造成网络环路！

修改配置文件/etc/river.yaml：

```
#### 控制器 IP
controller-ips:
  - 172.20.1.128
enable-profiler: false
#### river 从该网卡接受 vxlan 流量
interfaces:
  - p4p2
#### river 将 vxlan 解封装后，将还原的流量发送到该网卡
output-interfaces:
  - tap0
```

启动 river：

```
systemctl enable river
systemctl start river
```

部署安装 - *rosen*

注意： 注意：Rosen 暂不支持对 IPv6 隧道的解封装。

服务器规格

推荐配置： - CPU：16 个逻辑核 - 内存：16G 内存 - 磁盘：50G 系统盘 - 控制面网口：1xGE - 数据面网口：2x10GE（一进一出）

上述配置下的物理服务器性能参数： - 10Mpps / 10Gbps

支持的物理网卡驱动： - Intel ixgbe* drivers (支持 Intel 82599 系列 10G 网卡) - Intel i40e driver (不支持 Intel X722 10Gb, Intel X710 10Gb 网卡) - Intel ixgbev driver - Intel ixgbev driver - Mellanox mlx4 driver - Mellanox mlx5 driver - Netronome nfp driver - QLogic (Cavium) qed* drivers - Broadcom bnxt_en driver - Cavium thunderx driver

部署 *Rosen*

- 队列设置

如果网卡使用的是 82599 系列 10G 网卡执行下面操作，否则跳过此项设置。

- 若部署机器的 CPU 数量大于 48，需要 BIOS 关闭超线程，否则不需要设置。

- 执行完上面步骤之后，若 CPU 数量大于 32，需要设置出接口队列的数量：

```
ethtool -L <出接口名字> combined <queue_number>
```

其中 queue_number = 62 - cpu 数量

- 安装包获取

- 如果使用 salt 部署，从控制器 /usr/local/deepflow/yum/7/ 获取安装包
- 如果使用 docker 部署，从 ISO 中的 extra-artifacts 目录下寻找相应的安装包

- centos 7

- 更新内核

安装 rosen 的节点机器需要 4.19 内核，如果非 4.19 内核需要找到 kernel-4.19.*.el7.x86_64.rpm 安装包进行内核更新，安装命令如下：

先把旧版本删除，然后安装新版本。

```
/usr/bin/rpm -e kernel-4.19.*-*.el7.x86_64 --nodeps
yum --disablerepo=* localinstall -y ./kernel-*.rpm
```

安装完内核之后重启服务器，确保在 4.19.17 内核中运行 rosen。

- 找到 rosen-*.el7.*.rpm, bpftool*.rpm, tcpdump*.rpm, libpcap*.rpm 复制到分发节点上(复制到/tmp 目录下)：
- 安装 rosen 以及依赖项

```
yum --disablerepo=* localinstall -y /tmp/{rosen,bpftool,tcpdump,libpcap}*.rpm
```

- centos 8

- 找到 rosen-*.el8.*.rpm 复制到分发节点上(复制到/tmp 目录下)：
- 安装 rosen

```
yum --disablerepo=* localinstall -y /tmp/rosen-*.el8.*.rpm
```

- 修改配置文件 /etc/rosen.conf

```
global_defs {

    # 日志文件
    log_file      /var/log/rosen.log
```

```

# mtu < 3050
mtu 1500

# 模式设置
# 0: DRV 高性能模式，需要网卡支持，详细支持列表参考用户手册
# 1: SKB 普通模式，对网卡没有要求
mode 0

# 控制器 IP:Port，可以填写多个 如：10.30.1.128:20035,10.33.49.29:3
000,10.33.49.23:5000
# 用逗号分割，注意逗号两边不要有空格。
env_sync_addr 10.30.1.128:20035

# 控制器上报时间间隔 unit: second (秒)
env_sync_period 60

# 请求超时时间 unit : millisecond (毫秒)
sync_timeout 10000
}

```

```

# 接口定义
interface_defs {

# 接口总数量 < 64

# 配置遵循出口写在前面，入口写在后面的顺序。

# 出口配置
<init> interface p2p1 {
    # 接口不进行转发，转发的目的端口使用 'none'
    int_fwd none
}

# 入口配置
<init> interface p2p2 {
    # 接口转发的目的接口，注意：接口都是一对一转发。
    int_fwd p2p1
}
}

```

- 运行

- systemctl enable rosen.service
- systemctl start rosen.service
- 查看版本使用：rosen -v

在虚拟机中部署

注意： 这里使用出口为 veth 设备的方式，把数据解封装到虚拟机中进行处理。

- 入口使用驱动为 virtio_net driver 虚拟网卡。可使用 ethtool 查看设备驱动名称，如：ethtool -i eth0
- 出口使用 vethpair 网络设备。执行下面命令创建 veth 设备，并追加到 /sbin/ifup-local 中，使其机器重启时能够创建。

创建 vethpair 设备

```
ip link add name veth0 type veth peer name veth1
ip link set dev veth0 up
ip link set dev veth1 up
```

对/sbin/ifup-local 设置一下权限：

```
chmod 755 /sbin/ifup-local
```

- 修改配置文件 /etc/rosen.conf, 例如：

```
global_defs {
    ... ..
    # 选用 SKB 模式
    mode 1
    ... ..
}

interface_defs {
    # 出口配置
    <init> interface veth0 {
        int_fwd    none
    }

    # 入口配置
    <init> interface eth0 {
        int_fwd    veth0
    }
}
```

注意： 注意：① 模式选用 1，使用 mode 1 ② 可使用 tcpdump -i veth1 -ne 查看转换后的数据。

如何调试

日志文件： /var/log/rosen.log

抓包调试： rosen 使用 XDP 收发包，网卡不再支持 tcpdump。额外提供 rosen-dump 用于抓包调试。

```
rosen-dump -i interface [OPTS]
```

```
OPTS:
-h          help
-x          Show packet payload
-w file     Write the raw packets to file rather than parsing and printing them out.
```

注意：运行 rosen-dump，确保禁用 SELinux

```
setenforce 0
```

云平台对接

软件包获取

基础功能的软件包默认在 iso 中，仅携带文件录入同步和采集器资源同步功能；其他可以在 iso 同级目录的云平台对接软件包中获取。

类型	云平台	软件包
闭源私有云	腾讯专有云	cloud-agent-tencent-tce
	阿里专有云	cloud-agent-apsara-stack
	华为私有云	cloud-agent-huawei-private
	青云私有云	cloud-agent-qingcloud
	金山银河云	cloud-agent-kingsoft-private
	微软 ACS	cloud-agent-microsoft-acs
	微软 ACS-CMDB	cloud-agent-cmb-cmdb
	vSphere	cloud-agent-vsphere
	平安云	cloud-agent-pingan
	NSP	cloud-agent-nsp
开源私有云	OpenStack	cloud-agent-openstack
	ZStack	cloud-agent-zstack
公有云	腾讯公有云	cloud-agent-tencent
	阿里公有云	cloud-agent-aliyun
	AWS	cloud-agent-aws
	青云公有云	cloud-agent-qingcloud

	微软公有云	cloud-agent-azure
	华为公有云	cloud-agent-huawei
	百度公有云	cloud-agent-baidu-bce
其他	F5 BigIP	cloud-agent-f5

通用配置

警告： 对接云平台时，需确保 deepflow 主备控制器 DNS 服务可以正常解析。

```
common:
  task_interval: 300
  dns_server:
    dns_port: 53
    dns_address:
      # - '114.114.114.114'
  listen_port: 20230
  statsd_port: 20040
  tencent_host: '127.0.0.1'
  hostname_to_ip_file: '/etc/hostname_to_ip.csv'
  api_timeout: 60
  log_level: info
```

字段	含义
task_interval	是所有轮询操作的间隔时间，这个值不能太小，如果间隔太小会导致上次的任务没有完成，然后又开始了新的任务，一般不需要修改
dns_port	DNS 服务器的端口
dns_address	DNS 服务器的地址，最多支持两个，如果不想启用 DNS 就不要填写地址，这样效率更高，且避免从 DNS 获取错误的 ip
listen_port	cloud-agent 服务所在的端口
statsd_port	连接 influxdb 的端口
tencent_host	腾讯云主机的 ip 地址，一般不需要改
hostname_to_ip_file	应用于 openstack 主机获取到的 ip 重复或错误的场景。和 DNS 不能同时配置。使用 mt host.list 查看有无重复的 ip，如果有则此配置置为 true。
api_timeout	cloud-agent 调用各云平台 API 的超时时间，默认 60，单位：s
log_level	cloud-agent 的日志级别，支持 debug、info、warning 三种，默认 info

区域配置

regions:

- name: region-1
- uuid: 321289d9-d3e9-40f6-8694-ef4d81028b0f

字段 含义

name 此文件里全局的配置项，region 的名称

uuid 此文件里全局的配置项，当云平台中 region_uuid 配置为这个 uuid 时，此云平台所有资源都属于这个 region

- 以下情况需要编辑 regions 信息
 - 不同区域间网络不通时，需要根据实际网络情况将云平台划分到不同区域，便于从控制器和采集器通信
 - 需要监控区域间流量时，需要根据实际规划将云平台划分到不同区域，保证流量统计和展示的准确性
- 将配置文件中各云平台的 region_uuid 按需填写为上述 uuid 时，就可以完成云平台的区域调整
- 使用 uuidgen 可生成随机 uuid

建议： 调整全局及各云平台的 region 配置后，如果系统默认区域、系统默认可用区无需使用，可用如下命令删除

```
mt default-az.delete
```

```
mt default-region.delete
```

- 创建/etc/hostname_to_ip.csv（文件位置可以在上述配置项中修改）
 - 文件格式: csv
 - 内容要求: 每行有且仅有一个逗号，逗号前为域名或主机名，逗号后面为合法 ip 地址
 - 文件说明: 代码会自动过滤文件中的空行
 - 文件内容示例:

```
a.domain.com,192.168.2.2
b.domain.com,192.168.2.3
c.domain.com,192.168.2.4
```
- 对接以下云平台后，需要将 FloatingIP/浮动 IP/弹性 IP 所在的网段在资源管理中手动录入为子网, 否则会导致 IP 与 VM 的关联信息无法展示
 - OpenStack
 - ZStack
 - 华为私有云
 - 华为公有云

- 青云私有云
- 青云公有云
- 腾讯云
- AWS
- 阿里云

FAQ

- 支持同时对接多个云平台，在进行对接操作时保证数据库无脏数据
 - 如果需要重新初始化数据库，进入数据库后执行以下命令


```
mysql> source /usr/local/deepflow/db/init.sql
```
- 学习不到云平台信息
 - cloud-agent 服务是否正常


```
# systemctl status cloud-agent.service
```

如果有 ConnectionError，请参考“部署 cloud Agent”章节，检查配置是否正确

```
... HTTPConnectionPool(host='172.16.1.200', port=35357) ...
... HTTPConnectionPool(host='controller', port=9696) ...
```
 - 检查 exchange 服务是否正常


```
# systemctl status exchange.service
```

如果有 ConnectionError，请检查 cloud-agent 服务是否正常

```
... HTTPConnectionPool(host='127.0.0.1', port=20230) ...
```
 - 查看 `http://127.0.0.1:20230/v1/info` 接口中是否有 networks、ports、vms 数据，如果没有则 exchange 不学习
- 云平台学习过程
 - cloud-agent 把各个云平台的数据汇聚到一起，通过 `http://127.0.0.1:20230/v1/info` 接口把数据提供出去
 - exchange 调用上述接口来获取云平台数据，exchange 根据每条数据的 id 来确定学习和删除数据
- 部分数据学习不到的情况
 - 录入的硬件设备部分信息学习不到：大多数是 yaml 文件的数据格式不对，此时去看 cloud-agent 的日志或对比示例数据

- vSphere 平台的部分 IP 学习不到：需要去检查虚机是否安装了 vmware tools 工具
- OpenStack 的部分 router 学习不到：在 cloud-agent 日志中去搜索 router 的名字，是否被过滤掉了
- 页面上的数据没有及时同步：可能是页面的缓存，或 api 的缓存，需要用 mt 命令看下数据是否同步到了数据库，如果 mt 命令里有则是缓存的问题，如果没有请看日志
- 其他资源学习不到：首先调用 `http://127.0.0.1:20230/v1/info` 接口，看返回的数据里是否有，如果没有且日志中也没有相关信息可能是代码 bug，如果有则去看 exchange 日志，如果日志里没有相关信息，可能是 exchange 的 bug
- host 目前不支持更新，host 分为两部分，但可能会出现删除再添加的情况
 - 一部分是计算节点的 host，计算节点的 host 是通过 api 获取的，id 不会变，但 ip 和 name 可能会变，这是不支持更新的
 - 一部分是网络节点的 host，网络节点的 host 是从 router 上获取的 name，但是 ip 不知道，要通过 DNS 或 CSV 文件或 hash 来获取 ip，然后通过 ip 来生成 id，所以 ip 变了 id 也就变了，会删除旧的 id，新增新的 id
 - 至于网络节点为什么 ip 生成 id，不用 name 生成 id，因为 name 也可能变化，和 ip 一样
- exchange 日志中发现同一个 vif 删除两次，内容如下

```
DEBUG resource.update_res_in_db.1328: REMOVE vif: {'lcuuid': '40a30773-e56c-55fa-bd83-cbb0f887f187'}
DEBUG resource.update_res_in_db.1342: success
DEBUG resource.update_res_in_db.1328: REMOVE vif: {'lcuuid': '40a30773-e56c-55fa-bd83-cbb0f887f187'}
WARNING resource._get_vif_id_by_uuid.539: [memory expire] vif(40a30773-e56c-55fa-bd83-cbb0f887f187) not found
ERROR resource._get_vif_id_by_uuid.545: Vif (40a30773-e56c-55fa-bd83-cbb0f887f187) not found
```

这种情况说明 cloud-agent 学习到了 id 重复的资源，一般是来自录入的硬件设备，排查重复资源过程如下：

- 在 exchange 的日志文件里去搜 40a30773-e56c-55fa-bd83-cbb0f887f187 这个 id，就会找到是何种资源
- 然后去硬件设备的 yaml 文件里去搜索，就能找到重复的资源 建议：录入硬件设备的同事需要必要的脚本先去 yaml 文件查找一遍有无重复的资源，来避免出现类似的问题后才发现

私有云

OpenStack

前提条件

- 支持的 OpenStack 版本
 - keystone: v2/v3
 - nova: v2.x
 - neutron: v2.0

用到的 OpenStack 中的 API

- keystone:
 - http://ip:35357/v2.0/tokens
 - http://ip:35357/v2.0/tokens/<token>/endpoints
 - http://ip:35357/v2.0/tenants
 - http://ip:35357/v2.0/tenants/<tenant_id>/users
 - http://ip:35357/v2.0/users
 - http://ip:35357/v3/auth/tokens
 - http://ip:35357/v3/services
 - http://ip:35357/v3/endpoints
 - http://ip:35357/v3/domains
 - http://ip:35357/v3/projects
 - http://ip:35357/v3/users
- neutron:
 - http://ip:9696/v2.0/floatingips
 - http://ip:9696/v2.0/ports
 - http://ip:9696/v2.0/networks
 - http://ip:9696/v2.0/subnets
 - http://ip:9696/v2.0/routers
 - http://ip:9696/v2.0/routers/<router_id>/l3-agents
 - http://ip:9696/v2.0/security-groups
- nova:
 - http://ip:8774/v2/<tenant_id>/servers/detail?all_tenants=true
 - http://ip:8774/v2/<tenant_id>/os-hypervisors
 - http://ip:8774/v2/<tenant_id>/os-hypervisors/

举例

配置项	填写内容	备注
云平台名称	例：openstack-1	云平台的名称，不可和其他云平台的名称重复。不支持中文名称，最大长度 64

所属区域	在候选框中选择区域，可不选	选择区域后表示云平台属于这个 region ，如果不选表示学习此云平台上的区域
资源同步控制器	在候选框中选择资源同步控制器，可不选	选择资源同步控制器表示云平台在这个控制器上同步资源，不选会在区域的主节点上同步资源，如果没有区域会在主控制器上同步资源
管理员用户名	例：admin	OpenStack 管理员用户名，用于 OpenStack API 调用权限认证（客户提供）
管理员密码		OpenStack 管理员密码，用于 OpenStack API 调用权限认证（客户提供）
管理员 VPC	例：admin	OpenStack 管理员租户，用于 OpenStack API 调用权限认证（客户提供）
keystone API 版本	例：V2	可选值 V2 和 V3
管理员域名	例：default	OpenStack 管理域名，用于 OpenStack API 调用权限认证，仅在 KeyStone v3 时生效（客户提供）
开启多域	例：否	是否有多个域，没有请置为否，可选值是/否（客户提供）
API endpoint 类型	例：public	OpenStack API 端点类型，用于访问 OpenStack 服务入口点的抉择，可选值 admin/public/internal（客户提供）
keystone URL	例： http://172.16.1.200:35357/v2.0	KeyStone 服务地址，用于 OpenStack API 调用权限认证（客户提供）
nova URL	例：http://172.16.1.200:8774/v2	当通过 keystone URL 不能获取 nova 的 endpoint 时，需要配置 nova 服务地址，用于获取宿主机、虚拟机等资源
neutron URL	例：http://172.16.1.200:9696	当通过 keystone URL 不能获取 neutron 的 endpoint 时，需要配置 neutron 服务地址，用于获取网

指定网络类型 例：默认类型

额外对接路由接口 匹配接口名称的正则表达式，格式参考页面默认值

宿主机 IP 获取方式 例：API 对接

排除宿主机 例：172.16.1.201

排除 VPC 例：service

排除 IP 例：fe80::0/64

络、端口、IP 等资源

可选值为默认类型和外网。默认类型表示根据 OpenStack 平台的网络类型来学习；外网类型则表示网络都学习为外网类型。

用于匹配接口名称的正则表达式，作用是将节点上的其他功能性 IP 同步到 DeepFlow 中

可选值为 API 对接和读取配置文件。读取配置文件表示宿主机的 ip 从/etc/hostname_to_ip.csv 读取

需要过滤的宿主机的 ip，可以配置多个，不支持正则

需要过滤的项目的名称，可以配置多个，支持正则

需要过滤的 ip/网段，可以配置多个，不支持正则

[OpenStack 控制器上查看 Keystone, Nova 及 Neutron 的 endpoint](#)

```
openstack endpoint show keystone
openstack endpoint show neutron
openstack endpoint show nova
```

[OpenStack 控制器上查看 Domain 的 name](#)

```
openstack domain list
```

Endpoint 使用域名时需在 DeepFlow 控制器做域名解析，编辑/etc/hosts 文件，添加到该文件的“managed zone”之外。应尽量使用 IP 地址而非域名，除非域名是可以依靠 DNS 解析的。

```
#####--user defined zone--
10.25.0.2 controller
```

[对接限制](#)

- 如果没有安装 L3 Agent 服务，则路由器不能学习到真实的 Launch Server IP

[vSphere](#)

[前提条件](#)

- 支持的 vSphere 版本：
 - 5.1
 - 5.5
 - 6.0

举例

配置项	填写内容	备注
云平台名称	例：vsphere-1	云平台的名称，不可和其他云平台的名称重复。不支持中文名称，最大长度 64
所属区域	在候选框中选择区域，可不选	选择区域后表示云平台属于这个 region，如果不选表示学习此云平台上的区域
资源同步控制器	在候选框中选择资源同步控制器，可不选	选择资源同步控制器表示云平台在这个控制器上同步资源，不选会在区域的主节点上同步资源，如果没有区域会在主控制器上同步资源
vCenter Server IP	例：10.30.20.172	vSphere 提供服务的 ip（客户提供）
vCenter Server 服务端口	例：443	vSphere 提供服务的端口，一般为 443（客户提供）
管理员用户名	例：root	vSphere 的登陆用户名，用于 cloud-agnet 服务获取数据（客户提供）
管理员密码		vSphere 的登陆密码，用于 cloud-agnet 服务获取数据（客户提供）
子网 IPv4 地址掩码	例：24	vSphere 的 IPv4 的子网掩码，默认 24，为虚拟机自动分配网段
子网 IPv6 地址掩码	例：64	vSphere 的 IPv6 的子网掩码，默认 64，为虚拟机自动分配网段
排除宿主机	例：172.16.1.201	需要过滤的宿主机的 ip，可以配置多个，不支持正则
排除 VPC	例：service	需要过滤的项目的名称，可以配置多个，支持正则
排除 IP	例：fe80::0/64	需要过滤的 ip/网段，可以配置多个，不支持正则

ZStack

前提条件

- 支持的版本：
 - 企业版 3.1.1
 - 企业版 3.4.0
 - 混合云版 3.5.0.1

用到的 ZStack 中的 API

- /zstack/v1/accounts/resources/refs
- /zstack/v1/hosts
- /zstack/v1/vm-instances/appliances/virtual-routers
- /zstack/v1/l3-networks
- /zstack/v1/vm-instances
- /zstack/v1/eips
- /zstack/v1/accounts

举例

配置项	填写内容	备注
云平台名称	例：zstack-1	云平台的名称，不可和其他云平台的名称重复。不支持中文名称，最大长度 64
所属区域	在候选框中选择区域，可不选	选择区域后表示云平台属于这个 region，如果不选表示学习此云平台上的区域
资源同步控制器	在候选框中选择资源同步控制器，可不选	选择资源同步控制器表示云平台在这个控制器上同步资源，不选会在区域的主节点上同步资源，如果没有区域会在主控制器上同步资源
用户名	例：admin_user	管理员的用户名（客户提供）
密码		管理员的密码（客户提供）
URL	例：http://127.0.0.1:8080	获取 ZStack 资源的 URL（客户提供）
额外对接路由接口	匹配接口名称的正则表达式，格式参考页面默认值	用于匹配接口名称的正则表达式，作用是将节点上的其他功能性 IP 同步到 DeepFlow 中

云杉 NSP

前提条件

- 支持的版本
 - 5.5.6
 - 5.5.5-R

用到的 NSP 中的 API

- /auth/login
- /api/orchestration/v1/az
- /api/orchestration/v1/host
- /api/orchestration/v1/logical-network
- /api/orchestration/v1/vfirewall?with_link=true
- /api/orchestration/v1/vgateway?with_link=true

- /api/orchestration/v1/vrouter-bridge?with_link=true
- /api/orchestration/v1/logical-router?with_link=true
- /api/orchestration/v1/logical-switch?with_network=true
- /api/orchestration/v1/vm

举例

配置项	填写内容	备注
云平台名称	例：nsp-1	云平台的名称，不可和其他云平台的名称重复。不支持中文名称，最大长度 64
所属区域	在候选框中选择区域，可不选	选择区域后表示云平台属于这个 region，如果不选表示学习此云平台上的区域
资源同步控制器	在候选框中选择资源同步控制器，可不选	选择资源同步控制器表示云平台在这个控制器上同步资源，不选会在区域的主节点上同步资源，如果没有区域会在主控制器上同步资源
用户名	例：admin_user	用于调用 API 进行认证的用户名（客户提供）
密码		用于调用 API 进行认证的密码（客户提供）
URL	例：https://127.0.0.1	访问 NSP 的 API 的 URL（客户提供）
额外对接路由接口	匹配接口名称的正则表达式，格式参考页面默认值	用于匹配接口名称的正则表达式，作用是将节点上的其他功能性 IP 同步到 DeepFlow

额外说明

- 目前未对接资源
 - 租户与 VPC 对应关系
 - 应对方案：页面手动绑定租户与 VPC 对应关系
 - 专线及网段
 - 应对方案：页面录入子网和网段信息
 - 对等连接
 - 应对方案：页面录入对等连接
 - 华为防火墙 vsys
 - BWT

华为 FusionSphere

前提条件

- 支持的 FusionSphere 版本
 - 6.3.1

- 支持的 ManageOne 版本
 - 6.3.1
 - 8.0.0

用到的华为云中的 API

- 原生 openstack 的 API
 - <https://ip:35357/v3/auth/tokens> (ManageOne 6.3.1 OTenantSecurity 服务 API 参考 01.docx)
 - <https://ip:35357/v3/services> (ManageOne 6.3.1 OTenantSecurity 服务 API 参考 01.docx)
 - <https://ip:35357/v3/endpoints> (ManageOne 6.3.1 OTenantSecurity 服务 API 参考 01.docx)
 - <https://ip:35357/v3/auth/domains> (ManageOne 6.3.1 OTenantSecurity 服务 API 参考 01.docx)
 - <https://ip:35357/v3/projects> (ManageOne 6.3.1 OTenantSecurity 服务 API 参考 01.docx)
 - <https://ip:35357/v3/users> (ManageOne 6.3.1 OTenantSecurity 服务 API 参考 01.docx)
 - <https://ip:9696/v2.0/floatingips> (FusionSphere Service 6.3.1 弹性 IP API 参考 02.docx)
 - <https://ip:9696/v2.0/ports> (FusionSphere Service 6.3.1 虚拟私有云 API 参考 04.docx)
 - <https://ip:9696/v2.0/networks> (FusionSphere Service 6.3.1 虚拟私有云 API 参考 04.docx)
 - <https://ip:9696/v2.0/subnets> (FusionSphere Service 6.3.1 虚拟私有云 API 参考 04.docx)
 - <https://ip:9696/v2.0/routers> (FusionSphere Service 6.3.1 虚拟私有云 API 参考 04.docx)
 - <https://ip:9696/v2.0/routers/l3-agents>
 - <https://ip:8774/v2/servers/detail> (FusionSphere Service 6.3.1 弹性云服务器 API 参考 02.docx)
- 用到的 ManageOne 中的 API
 - <https://ip:26335/rest/plat/smapp/v1/oauth/token> (ManageOne 6.3.1 运维面 API 参考 03.docx)
 - https://ip:26335/rest/tenant-resource/v1/tenant/resources/CLOUD_HOST (ManageOne 6.3.1 运维面 API 参考 03.docx)
 - https://ip:26335/rest/cmdb/v1/instances/SYS_PhysicalHost (ManageOne 6.3.1 运维面 API 参考 03.docx)

- https://ip:26335/rest/tenant-resource/v1/tenant/resources/CLOUD_VM (ManageOne 6.3.1 运维面 API 参考 03.docx)
- https://ip:26335/rest/tenant-resource/v1/tenant/resources/TENANT_VDC (ManageOne 6.3.1 租户资源北向模型 01.xlsx)
- https://ip:26335/rest/tenant-resource/v1/tenant/resources/CLOUD_ROUTER (ManageOne 6.3.1 租户资源北向模型 01.xlsx)
- https://ip:26335/rest/tenant-resource/v1/tenant/resources/CLOUD_NETWORK (ManageOne 6.3.1 租户资源北向模型 01.xlsx)
- https://ip:26335/rest/tenant-resource/v1/tenant/resources/CLOUD_SUBNET (ManageOne 6.3.1 租户资源北向模型 01.xlsx)
- https://ip:26335/rest/tenant-resource/v1/tenant/resources/CLOUD_PORT (ManageOne 6.3.1 租户资源北向模型 01.xlsx)
- https://ip:26335/rest/tenant-resource/v1/instances/relations/R_PORT_IN_SUBNET (ManageOne 6.3.1 租户资源北向模型 01.xlsx)

举例

配置项	填写内容	备注
云平台名称	例：huawei-private-1	云平台的名称，不可和其他云平台的名称重复。不支持中文名称，最大长度 64
所属区域	在候选框中选择区域，可不选	选择区域后表示云平台属于这个 region，如果不选表示学习此云平台上的区域
资源同步控制器	在候选框中选择资源同步控制器，可不选	选择资源同步控制器表示云平台在这个控制器上同步资源，不选会在区域的主节点上同步资源，如果没有区域会在主控制器上同步资源
使用 OpenStack 原生 API (1)	在候选框中选择是或否	是否使用 opesntack 原生的 API，false 表示只使用 manageOne 的 API
管理员用户	例：admin	OpenStack 管理员用户名，用

名 (2)		于 OpenStack API 调用权限认证 (客户提供)
管理员密码 (3)		OpenStack 管理员密码, 用于 OpenStack API 调用权限认证 (客户提供)
管理员 VPC (4)	例: admin	OpenStack 管理员租户, 用于 OpenStack API 调用权限认证 (客户提供)
管理员域名 (5)	例: default	OpenStack 管理域名, 用于 OpenStack API 调用权限认证, 仅在 Keystone v3 时生效 (客户提供)
开启多域 (6)	例: 否	是否有多个域, 没有请置为否, 可选值是/否 (客户提供)
API Endpoint 类型 (7)	例: public	OpenStack API 端点类型, 用于访问 OpenStack 服务入口点的抉择, 可选值 admin/public/internal (客户提供)
Keystone URL (8)	例: http://172.16.1.200:35357/v2.0	Keystone 服务地址, 用于 OpenStack API 调用权限认证 (客户提供)
指定网络类型 (9)	例: 默认类型	可选值为默认类型和外网。默认类型表示根据 OpenStack 平台的网络类型来学习; 外网类型则表示网络都学习为外网类型。
项目作为 VPC (10)	在候选框中选择是或否	否表示 VDC 作为 VPC, 是表示项目作为 VPC
ManageOne 运维侧用户名 (11)	例: admin_user	登陆 ManageOne 页面: 系统管理 -> 用户管理 页面查看, 用户类型为: 三方系统接入用户 (客户提供)
ManageOne 运维侧密码 (12)		登陆 ManageOne 页面: 系统管理 -> 用户管理 页面用户的密码 (客户提供)
ManageOne 运维侧 URL (13)	例: https://10.10.39.100:26335	用于调用 ManageOne 运维侧的 API: URL 的 IP 一般是 ManageOne 页面的 IP, 端口一

额外对接路由接口 (14)	匹配接口名称的正则表达式，格式参考页面默认值	一般是 26335（客户提供） 用于匹配接口名称的正则表达式，作用是将节点上的其他功能性 IP 同步到 DeepFlow 中
区域白名单 (15)	例：区域名称 1	可以配置多个，不支持正则表达式，多个区域名称之间以英文逗号分隔（客户提供）
可用区白名单 (16)	例：可用区名称 1,可用区名称 2	可以配置多个，不支持正则表达式，多个可用区名称之间以英文逗号分隔
VPC 白名单 (17)	例：VPC 名称 1,VPC 名称 2	可以配置多个，不支持正则表达式，多个 VPC 名称之间以英文逗号分隔

如果使用 OpenStack 原生 API 和 ManageOne 方式对接需要客户提供配置项 1-17；如果只使用 ManageOne 方式对接需要客户提供配置项 11-17（推荐只使用 ManageOne 对接的方式）

华为云控制器上查看 [Keystone](#)，[Nova](#) 及 [Neutron](#) 的 endpoint

```
openstack endpoint show keystone
openstack endpoint show neutron
openstack endpoint show nova
```

Endpoint 使用域名时需在 DeepFlow 控制器做域名解析，编辑/etc/hosts 文件，添加到该文件的“managed zone”之外。应尽量使用 IP 地址而非域名，除非域名是可以依靠 DNS 解析的。

```
#####--user defined zone--
10.25.0.2          controller
```

[腾讯专有云](#)

前提条件

- 5.6.1 版本开始支持

用到的腾讯云 SDK 中的 API

- DescribeRegions - 获取区域和可用区信息
- DescribeHosts - 获取宿主机信息
- DescribeVpcEx - 获取 VPC 信息
- DescribeVpcPeers - 获取对等连接信息
- DescribeBgRule - 获取负载均衡器信息
- DescribeNatGateways - 获取 NAT 网关信息
- DescribeRtbsEx - 获取路由表信息

- DescribeSubnetEx - 获取子网信息
- DescribeInstances - 获取云服务器信息
- DescribeNetworkInterfacesEx - 获取云服务器接口信息
- DescribeSecurityGroup - 获取安全组信息
- DescribeSecurityGroupPolicy - 获取安全组策略
- DescribeCvmForSecurityGroup - 获取安全组与云服务器关联关系
- DescribeCluster - 获取 TKE 集群信息
- QueryInstances - 补充云服务器及接口信息

举例

配置项	填写内容	备注
云平台名称	例: <code>tencent-tce-1</code>	云平台的名称, 不可和其他云平台的名称重复。不支持中文名称, 最大长度 64。
所属区域	在候选框中选择区域, 可不选	选择区域后表示云平台属于这个 region , 如果不选表示学习此云平台上的区域
资源同步控制器	在候选框中选择资源同步控制器, 可不选	选择资源同步控制器表示云平台在这个控制器上同步资源, 不选会在区域的主节点上同步资源, 如果没有区域会在主控制器上同步资源
AccessKey ID	例: <code>AKIDztZ0C9dHuIQJwxxxxxxigTBhz2LQks</code>	使用运营侧账号登录云管页面: 右上角平台管理 -> 用户与权限 -> 左侧云 API 秘钥, 查看 SecretId 客户提供
AccessKey Secret	例: <code>73lFWYLtLbgW7UcV8xxxxxxH3ZcvnPlx</code>	使用运营侧账号登录云管页面: 右上角平台管理 -> 用户与权限 -> 左侧云 API 秘钥, 查看 SecretKey 客户提供
API Endpoint	例: <code>yunapi3.yundev.xxxxxx.com</code>	运营侧 API Endpoint , 通常为 <code>yunapi3.\$main_domain</code> 的格式, 作为参考租户侧登录地址为

区域名称 例：地域为重庆，填写 chongqing

额外对接路由接口 匹配接口名称的正则表达式，格式参考页面默认值

排除区域 例：shanghai,shenzhen

console.\$main_domain、运营侧登录地址为 cmgt.\$main_domain 客户提供

AccessKey 有权限的任意一个地域英文名称，该值仅用于调用 API 获取区域的完整列表 客户提供

用于匹配接口名称的正则表达式，作用是将节点上的其他功能性 IP 同步到 DeepFlow 中

需要过滤的区域的名称，可以配置多个，不支持正则

AccessKey ID、Secret 获取方法（使用运营侧账号登录）：



腾讯 TCE AccessKey 获取方法

限制

- 公共服务 VPC、基础网络默认 VPC
 - 公共服务 VPC（VPC -1）中的 CVM 位于 Underlay 母机上，基础网络默认 VPC（VPC 0）中的 CVM 位于 Overlay 母机上
 - TCE 的 VPC GET API 不会返回这两个 VPC，但在云服务器等其他 API 中会有资源指向 VpcId -1 或 0
 - 这些 VPC 中的云服务器无法通过 TCE 运营侧 API 获取网卡信息，无法展示 MAC、IP 信息
- 云防火墙
 - 作为云防火墙使用的云服务器网卡会连接到多个 VPC 的子网，目前仅支持将连接到云服务器所属 VPC 的网卡同步过来
- 虚拟网卡
 - 网卡名前缀为 veth_ 的是云服务器的主网卡，前缀为 veni_ 是其他网卡
 - 网卡 MAC 信息按照以下优先级确定：

- 如果 TCE 运营侧 API 返回了 MAC，直接使用 API 的 MAC
- 如果采集器返回了该接口则使用采集器的信息：veth_开头的网卡 mac 前缀替换为 52，veni_开头的网卡 mac 前缀替换为 20
- 其他网卡，veth_前缀的使用 52:54:00 前缀与网卡名拼接
- NAT 网关
 - NAT 规则不支持同步（TCE 运营侧 API 无法正常调用）
- 安全组
 - 安全组和虚拟机的关联关系不支持同步（TCE 运营侧 API 错误概率太高）

阿里专有云

前提条件

- 5.6.1 版本开始支持
- 支持的阿里专有云版本
 - 3.8.1
 - 运维平台：DT 2.x
 - 3.11+
 - 运维平台：ASCM 3.x

阿里专有云使用到的 API

- DescribeRegions
- DescribeZones
- DescribeVpcs
- DescribeNatGateways
- DescribeSnatTableEntries
- DescribeForwardTableEntries
- DescribeRouteTableList
- DescribeRouteEntryList
- DescribeVSwitches
- DescribeSecurityGroups
- DescribeSecurityGroupAttribute
- DescribeInstances
- DescribeNetworkInterfaces
- DescribeLoadBalancers
- DescribeLoadBalancerAttribute
- DescribeHealthStatus
- DescribeDBInstances
- DescribeDBInstanceAttribute
- DescribeDBInstanceNetInfo

举例

配置项	填写内容	备注
云平台名称	例：apsara-stack-1	云平台的名称，不可和其他云平台的名称重复。不支持中文名称，最大长度 64。
所属区域	在候选框中选择区域，可不选	选择区域后表示云平台属于这个 region，如果不选表示学习此云平台上的区域
资源同步控制器	在候选框中选择资源同步控制器，可不选	选择资源同步控制器表示云平台在这个控制器上同步资源，不选会在区域的主节点上同步资源，如果没有区域会在主控制器上同步资源
AccessKey ID	例：LTAI4FiU3ad3txxxxxx8xGfn	登录 ASCM 页面：企业 -> 组织管理 -> 选择正确的组织 -> 获取 accesskey -> 查看 AccessKey ID 客户提供
AccessKey Secret	例： itsHzkPo22jbtxxxxxxz3gc5bsPnXP	登录 ASCM 页面：与 AccessKey ID 查询页面相同，查看 AccessKey Secret 即可 客户提供
API Endpoint	例：res.sgmc.xxxx.com.cn	登录天基运维页面：报表 -> 全部报表 -> 搜索服务注册变量 -> 搜索 asapi，获取它的 asapi.server.endpoint 值 客户提供
区域名称	例：beijing	登录 ASCM 页面：右上角个人信息 -> 找到一个区域名。若通过其他途径找到 AccessKey 有权限的任意一个区域也可，该值仅用于 API 获取区域的完整列表 客户提供
排除区域	例：shanghai,shenzhen	需要过滤的区域的名称，可以配置多个，不支持正则

青云私有云

前提条件

- 5.5.8 版本开始支持
- 客户必须安装 BOSS2 平台

青云私有云使用到的 API

- /iaas/?action=DescribeZones
- /iaas/?action=DescribeRouters
- /iaas/?action=DescribeRouteTables
- /iaas/?action=DescribeVpcBorders
- /iaas/?action=DescribeBorderVxnets
- /iaas/?action=DescribeRouterVxnets
- /iaas/?action=DescribeVxnets
- /iaas/?action=DescribeSecurityGroups
- /iaas/?action=DescribeSecurityGroupRules
- /iaas/?action=DescribeNFVs
- /iaas/?action=DescribeLoadBalancers
- /iaas/?action=DescribeBalancerListeners
- /iaas/?action=DescribeLoadBalancerBackends
- /iaas/?action=DescribeInstances
- /iaas/?action=DescribeNics
- /iaas/?action=DescribeEips
- /boss2/?action=Boss2DescribeBots

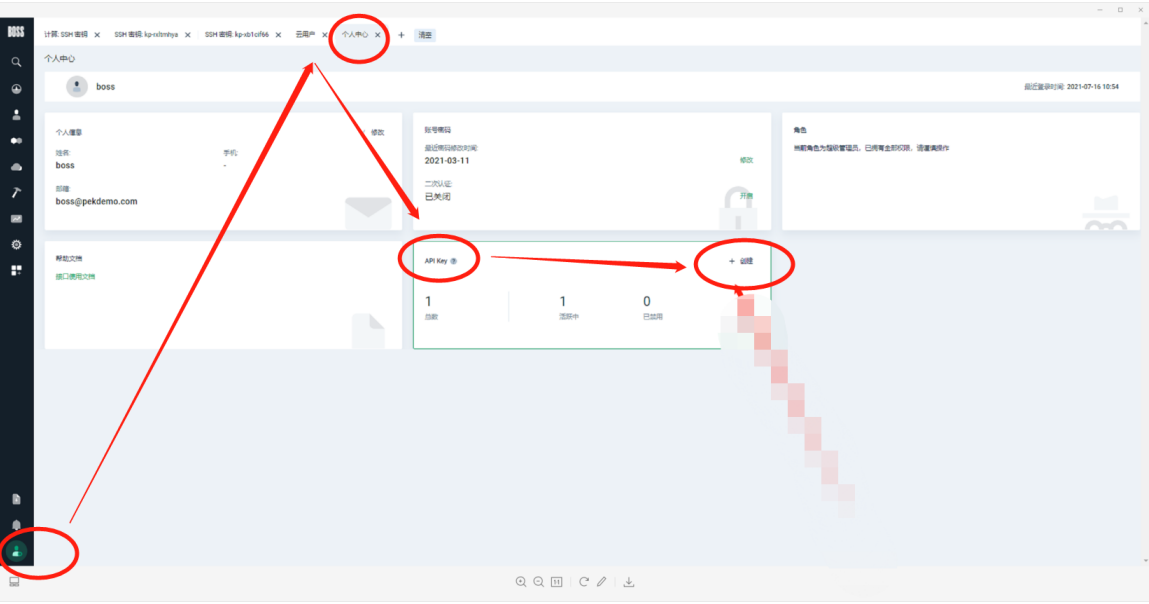
举例

配置项	填写内容	备注
云平台名称	例：qingcloud-private-1	云平台的名称，不可和其他云平台的名称重复。不支持中文名称，最大长度 64。
所属区域	在候选框中选择区域，可不选	选择区域后表示云平台属于这个 region，如果不选表示学习此云平台上的区域
资源同步控制器	在候选框中选择资源同步控制器，可不选	选择资源同步控制器表示云平台在这个控制器上同步资源，不选会在区域的主节点上同步资源，如果没有区域会在主控制器上同步资源
青云 API 密钥 ID	例：VSXGWCXWKCxxxxxxWMAJ	青云页面 主页面右上角-API 密钥-API 密钥管理-API 密钥 ID（客户提供）
青云 API 密钥 KEY	例： 9gifjABLeYWIT0xGWD1DxxxxxxzS1sHu7e	青云 API 密钥 ID 对应的 API 密钥 KEY（客户提供）
青云 API 入	例：https://api.qingcloud.com	青云调用 API 的 URL（客户提供）

BOSS 平台 API 密钥 ID	例：VSXGWCXWKCxxxxxxWMAJ	BOSS 平台的 API 密钥 ID (客户提供)
BOSS 平台 API 密钥 KEY	例： 9gifjABLeYWlTOxGWD1DxxxxxxzS1sHu7e	BOSS 平台的 API 密钥 KEY (客户提供)
BOSS 平台 API 入口	例：https://boss.cloud.xxxx.com	BOSS 平台调用 API 的 URL (客户提供)
额外对接路由接口	匹配接口名称的正则表达式，格式参考页面默认值	用于匹配接口名称的正则表达式，作用是将平台宿主主机上的其他功能性 IP 同步到 DeepFlow 中

BOSS 平台的 API 密钥获取方式

- 登录后点击页面左下角访问个人中心，创建一个 API Key，如下图所示：



获取方式

额外说明

- 负载均衡器、NAT 网关、路由器都存在具体的载体虚拟机

限制

- 区域
 - 不支持同步多区域，目前会把所有区域的资源统一归集到系统默认区域
- 子网
 - 不支持同步 DVR 模式的 vxnet
- 宿主机
 - 不支持同步关机状态的宿主机，进而影响同步上面的虚拟机（宿主机关机后，API 无法返回 IP 字段）
- 负载均衡器
 - 不支持同步后端主机与负载均衡器在不同可用区的负载均衡规则
- NAT 网关
 - 云平台本身没有 NAT 网关规则
 - NAT 网关关联路由表后，会自动增加指向 NAT 网关的路由表规则

金山银河云

前提条件

- 5.6.3 版本开始支持
- 仅支持 Keystone 2.0 版本

金山银河云使用到的 API

- Keystone:
 - `http://ip:35357/v2.0/tokens`
 - `http://ip:35357/v2.0/tokens/<token>/endpoints`
 - `http://ip:35357/v2.0/tenants`
 - `http://ip:35357/v2.0/tenants/<tenant_id>/users`
 - `http://ip:35357/v2.0/users`
- Neutron:
 - `http://ip:9696/v2.0/vpc/domains`
 - `http://ip:9696/v2.0/vpc/vnets`
 - `http://ip:9696/v2.0/floatingips`
 - `http://ip:9696/v2.0/routers`
 - `http://ip:9696/v2.0/routers/<router_id>/l3-agents`
 - `http://ip:9696/v2.0/security-groups`
- Nova:
 - `http://ip:8774/v2/<tenant_id>/servers/detail?all_tenants=true`
 - `http://ip:8774/v2/<tenant_id>/os-hypervisors`
 - `http://ip:8774/v2/<tenant_id>/os-hypervisors/`

举例

配置项	填写内容	备注
云平台名称	例：kingsoft-private-1	云平台的名称，不可和其他云平台的名称重复。不支持中文名称，最大长度 64。
所属区域	在候选框中选择区域，可不选	选择区域后表示云平台属于这个 region，如果不选表示学习此云平台上的区域
资源同步控制器	在候选框中选择资源同步控制器，可不选	选择资源同步控制器表示云平台在这个控制器上同步资源，不选会在区域的主节点上同步资源，如果没有区域会在主控制器上同步资源
管理员用户名	例：admin	青云页面 主页面右上角-API 密钥-API 密钥管理-API 密钥 ID（客户提供）
管理员密码	例：xxxxxxxxxxx	青云 API 密钥 ID 对应的 API 密钥 KEY（客户提供）
管理员 VPC	例：admin	青云调用 API 的 URL（客户提供）
开启多域	在候选框中选择是或否，默认否	是否有多个域
API endpoint 类型	在候选框中选择 admin 或 public 或 internal	OpenStack API 端点类型，用于访问 OpenStack 服务入口点的抉择
keystone URL	例： http://xxx.xxx.xxx.xxx:5000/v2.0	KeyStone 服务地址，用于 OpenStack API 调用权限认证（客户提供）
nova URL	例： http://xxx.xxx.xxx.xxx:8774/v2	当通过 auth_uri 不能获取 nova 的 endpoint 时，需要配置 nova 服务地址，用于获取物理机、虚拟机等资源
neutron URL	例：http://xxx.xxx.xxx.xxx:9696	当通过 auth_uri 不能获取 neutron 的 endpoint 时，需要配置 neutron 服务地址，用于获取网络、端口、IP 等资源
指定网络类型	在候选框中选择默认类型或外网类型	值为默认类型，则根据 OpenStack 平台的网络类型来学习；值为外网，则网络都学习为外网类型

宿主机 IP 获取方式	在候选框中选择 API 对接或读取配置文件	当通过 API 不能正确获取宿主机的 IP 时，选择读取配置文件后，需要配置好/etc/cloud-agent.yaml 里的 hostname_to_ip_file 配置项
排除宿主机	例：192.168.1.1,192.168.1.2	需要过滤的宿主机的 ip，可以配置多个，用英文逗号分隔，不支持正则，
排除 VPC	例：test-*	需要过滤的项目的名称，可以配置多个，用英文逗号分隔，仅支持正则
排除 IP	例：fe80::0/64	需要过滤的 IP，可以配置多个，用英文逗号分隔，需指定 IP 的网段

微软 ACS

前提条件

- 5.6.3 版本开始支持
- VMM 版本
 - 10.19.2445.0
- NC 版本
 - 12.0.56

微软 ACS 使用到的命令

- VMM 中使用的命令
 - Get-SCVMMServer
 - Get-SCVMHost （导出为 JSON 格式，结果保存在 VMHost.txt 文件中）
 - Get-SCLogicalNetworkDefinition （导出为 JSON 格式，结果保存在 VMHost-Network.txt 文件中）
 - Get-SCVirtualNetworkAdapter （导出为 JSON 格式，结果保存在 VMHost-Port.txt 文件中）
 - Get-ProviderAddress （导出为 JSON 格式，结果保存在 VMHost-ProviderAddress.txt 文件中）
 - Get-SCVMNetwork （导出为 JSON 格式，结果保存在 VMNetwork.txt 文件中）
 - Get-SCVirtualMachine （导出为 JSON 格式，结果保存在 VM.txt 文件中）
- NC 中使用的命令
 - 获取虚拟网络（VNet）（导出为 JSON 格式，结果保存在 VirtualNetwork.txt 文件中）

- 获取负载均衡（SLB、iLB）（导出为 JSON 格式，结果保存在 LB.txt 文件中）
- 获取网络安全组（NSG）（导出为 JSON 格式，结果保存在 NetworkSecurityGroup.txt 文件中）
- 获取外网 IP 地址（PIP）（导出为 JSON 格式，结果保存在 PublicIP.txt 文件中）

举例

配置

项	填写内容	备注
云平台名称	例：microsoft-acs-1	云平台的名称，不可和其他云平台的名称重复。不支持中文名称，最大长度 64。
所属区域	在候选框中选择区域，可不选	选择区域后表示云平台属于这个 region，如果不选表示学习此云平台上的区域
资源同步控制器	在候选框中选择资源同步控制器，可不选	选择资源同步控制器表示云平台在这个控制器上同步资源，不选会在区域的主节点上同步资源，如果没有区域会在主控制器上同步资源
资源文件目录	例：/etc/microsoft-acs/	把上述 txt 资源文件都放在这个目录下
宿主机管理网段	例： 10.33.1.0/24,10.33.0.0/16	登录 VMM 页面，查看宿主机管理 IP 所在的网段，多个网段用英文逗号隔开，用于确定宿主机的管理 IP
外网网段	例：10.33.0.0/16	只可配置一个网段，用于判断负载均衡是内部负载均衡还是外部负载均衡

额外配置

- 通过 API 目前无法获取 PIP 与云服务器的一一映射关系，PIP 的作用相当于云服务器的 Floating IP，需要将 PIP 的 CIDR 手动录入为 VIP=是、类型=外网的子网，且注意区域、可用区、VPC 信息与 PIP 所关联的云服务器一致
- 云上云下互访时，如果通过 Gateway 走路由直接通信，需要将云下 IP 所在的 CIDR 录入为类型=外网的子网，或者将云下 CIDR 录入为类型=内网的子网并手动建立云上云下 VPC 之间的对等连接

EasyStack

前提条件

- 支持的 EasyStack 版本

- v3
- v5
- v6（待定）

其他

- 参考 [OpenStack](#)

公有云

AWS

前提条件

- 支持的 api 版本：
 - 2016-11-15

用到的 AWS SDK 中的 API

- describe_vpcs
- describe_vpc_peering_connections
- describe_nat_gateways
- describe_internet_gateways
- describe_route_tables
- describe_subnets
- describe_security_groups
- describe_instances
- describe_network_interfaces

举例

配置项	填写内容	备注
云平台名称	例：aws-1	云平台的名称
所属区域	在候选框中选择区域，非必填	选择区域后表示云平台属于这个区域，否则属于系统默认区域
资源同步控制器	在候选框中选择资源同步控制器，非必填	选择资源同步控制器表示云平台在这个控制器上同步资源，否则会自动选择所属区域的主节点同步资源；如果没有指定所属区域则会自动选择从主控制器同步资源
AccessKey ID	例：AKIA22EJFA5HIQMSOBVS	在控制台页面 我的安全凭证-API 的访问密钥 的页面上的访问密钥 ID（客户提供）

AccessKey Secret	例: itsHzkPo22jbtNZ61QEz3gc5bsPnXP	只有第一次创建 secret_id 和 secret_key 的时候可以下载 secret_key (客户提供)
区域白名单	例: cn-north-1	可以配置多个, 不支持正则表达式; 优先级高于区域黑名单
区域黑名单	例: cn-northwest-1,cn-northeast-1	可以配置多个, 不支持正则表达式

阿里公有云

前提条件

- 5.5.7 版本开始支持

用到的 Python SDK 中的 API

- DescribeRegions
- DescribeVpcs
- DescribeNatGateways
- DescribeSnatTableEntries
- DescribeForwardTableEntries
- DescribeRouteTableList
- DescribeRouteEntryList
- DescribeVSwitches
- DescribeInstances
- DescribeNetworkInterfaces
- DescribeSecurityGroups
- DescribeSecurityGroupAttribute
- DescribeLoadBalancers
- DescribeLoadBalancerAttribute
- DescribeHealthStatus

举例

配置项	填写内容	备注
云平台名称	例: aliyun-1	云平台的名称
所属区域	在候选框中选择区域, 非必填	选择区域后表示云平台属于这个区域, 否则属于系统默认区域
资源同步控制器	在候选框中选择资源同步控制器, 非必填	选择资源同步控制器表示云平台在这个控制器上同步资源, 否则会自动选择所属区域的主节点同步资源; 如果没有指定

AccessKey ID	例: LTAI4FiU3ad3txLUSRg8xGfn	所属区域则会自动选择从主控制器同步资源 请在控制台-accesskeys 页面上配置用于 API 访问的密钥 ID (客户提供)
AccessKey Secret	例: itsHzkPo22jbtNZ61QEz3gc5bsPnXP	请在控制台-accesskeys 页面上配置用于 API 访问的密钥 KEY (客户提供)
区域白名单	例: 华北 3	可以配置多个, 不支持正则表达式; 优先级高于区域黑名单
区域黑名单	例: 华南 3 (广州), 华北 6 (乌兰察布)	可以配置多个, 不支持正则表达式

知名服务网段

建议录入以下网段 - 100.64.0.0/10: 所有 VPC 共用的云产品通信 - 100.100.2.136、100.100.2.138: DNS 服务器地址 - 100.100.5.1: NTP 服务地址

腾讯公有云

前提条件

- 支持的 api 版本:
 - 3.0

用到的腾讯云 SDK 中的 API

- DescribeRegions
- DescribeVpcs
- DescribeSubnets
- DescribeInstances
- DescribeRouteTables
- DescribeSecurityGroups
- DescribeSecurityGroupPolicies
- DescribeServiceTemplates
- DescribeAddressTemplates
- DescribeAddressTemplateGroups
- DescribeNetworkInterfaces
- DescribeAddresses

举例

配置项	填写内容	备注
云平台名	例: tencent-1	云平台的名称

称

所属区域 在候选框中选择区域，非必填

选择区域后表示云平台属于这个区域，否则属于系统默认区域

资源同步
控制器 在候选框中选择资源同步控制器，非必填

选择资源同步控制器表示云平台在这个控制器上同步资源，否则会自动选择所属区域的主节点同步资源；如果没有指定所属区域则会自动选择从主控制器同步资源

AccessKey
ID 例：
AKIDztZ0C9dHuIQJwKMeZEixykjTBhz2L

在腾讯云页面 访问管理-云API 密钥-API 密钥管理 页面上的 SecretId（客户提供）

AccessKey
Secret 例： itsHzkPo22jbtNZ61QEz3gc5bsPnXP

在腾讯云页面 访问管理-云API 密钥-API 密钥管理 页面上的 SecretKey（客户提供）

区域白名
单 例：华东地区(上海)

可以配置多个，不支持正则表达式；优先级高于区域黑名单

区域黑名
单 例：西南地区(重庆),西南地区(成都)

可以配置多个，不支持正则表达式

知名服务网段

建议录入以下网段 - 169.254.0.0/16：所有 VPC 共用的云产品通信

华为公有云

前提条件

- 5.5.8 版本开始支持

用到的华为公有云的 API

- /v3/auth/tokens
- /v3/regions
- /v3/auth/projects
- /v1/<project_id>/vpcs
- /v2.1/<project_id>/os-availability-zone
- /v2.0/vpc/peerings

- /v2.0/vpc/routes
- /v1/<project_id>/subnets
- /v1/<project_id>/security-groups
- /v1/<project_id>/cloudservers/detail
- /v1/<project_id>/ports
- /v2/<project_id>/nat_gateways
- /v2/<project_id>/dnat_rules
- /v2/<project_id>/snat_rules
- /v1/<project_id>/publicips
- /v2.0/lbaas/loadbalancers
- /v2.0/lbaas/listeners
- /v2.0/lbaas/pools/<pool_id>/members

举例

配置项	填写内容	备注
云平台名称	例： huawei-1	云平台的名称
所属区域	在候选框中选择区域，非必填	选择区域后表示云平台属于这个区域，否则属于系统默认区域
资源同步控制器	在候选框中选择资源同步控制器，非必填	选择资源同步控制器表示云平台在这个控制器上同步资源，否则会自动选择所属区域的主节点同步资源；如果没有指定>所属区域则会自动选择从主控制器同步资源
帐号名	例： yunshan_network	在华为云页面，访问控制台-我的凭证页面上找到 帐号名（客户提供）
IAM 用户密码		登录华为云页面的密码（客户提供）
项目ID	例： 08347ff64380261b2fb5c01a8c6f9835	控制台-我的凭证-项目列表页面上其中一个项目 ID（客户提供）
项目区域白名单	例： cn-east-3 例： 华东地区(上海)	项目 ID 对应的项目（客户提供） 可以配置多个，不支持正则表达式；优先级高于区域黑名单

区域黑名单

例：西南地区(重庆),西南地区(成都)

可以配置多个，不支持正则表达式

青云公有云

前提条件

- 5.5.8 版本开始支持

青云公有云使用到的 API

- /iaas/?action=DescribeZones
- /iaas/?action=DescribeRouters
- /iaas/?action=DescribeRouteTables
- /iaas/?action=DescribeVpcBorders
- /iaas/?action=DescribeBorderVxnets
- /iaas/?action=DescribeRouterVxnets
- /iaas/?action=DescribeVxnets
- /iaas/?action=DescribeSecurityGroups
- /iaas/?action=DescribeSecurityGroupRules
- /iaas/?action=DescribeNFVs
- /iaas/?action=DescribeLoadBalancers
- /iaas/?action=DescribeBalancerListeners
- /iaas/?action=DescribeLoadBalancerBackends
- /iaas/?action=DescribeInstances
- /iaas/?action=DescribeNics
- /iaas/?action=DescribeEips

举例

配置

项	填写内容	备注
云平台名称	例：qingcloud-1	云平台的名称
所属区域	在候选框中选择区域，非必填	选择区域后表示云平台属于这个区域，否则属于系统默认区域
资源同步控制器	在候选框中选择资源同步控制器，非必填	选择资源同步控制器表示云平台在这个控制器上同步资源，否则会自动选择所属区域的主节点同步资源；如果没有指定所属区域则会自动选择从主控

API 密钥 ID	例: AKIDztZ0C9dHuIQJwKMeZEixykjTBhz2L	制器同步资源 在青云页面 主页面右上角-API 密钥-API 密钥管理-API 密钥 ID (客户提供)
API 密钥 KEY	例: itsHzkPo22jbtNZ61QEz3gc5bsPnXP	API 密钥 ID 对应的 API 密钥 KEY (客户提供)

微软公有云

前提条件

- 5.6.1 版本开始支持

用到的 Python SDK 中的 API

- client.subscriptions.list_locations
- client.virtual_networks.list_all
- client.virtual_network_peerings.list
- client.nat_gateways.list_all
- client.route_tables.list_all
- client.network_security_groups.list_all
- client.virtual_machines.list_all
- client.network_interfaces.list_all
- client.public_ip_addresses.list_all

举例

配置项	填写内容	备注
云平台名称	例: azure-1	云平台的名称，不可和其他云平台的名称重复。不支持中文名称，最大长度64
所属区域	在候选框中选择区域，可不选	选择区域后表示云平台属于这个region，如果不选表示学习此云平台上的区域
资源同步控制器	在候选框中选择资源同步控制器，可不选	选择资源同步控制器表示云平台在这个控制器上同步资源，不选会在区域的主节点上同步资源，如果没有区域会在主控>制器上同步资源
租户ID	例: 85a117f0-b847-xxxx-ccc8-7b5e8f140401	主页 - 搜索 Azure Active Directory - 租户信息 - 租户 ID (客户提供)
应用程序(客户端)	例: a10917bb-3628-xxxx-82a0-2896715e094e	主页 - 搜索 Azure Active Directory - 应用注册 - 应用程序(客户端) ID (客户提供)

ID

客户端 例: N-
密码 J9_Qzoq~cadMxxxxapL9VU-
E.w.M9n

订阅 例: 99a21c9b-ec45-xxxx-ad08-
ID c316b16f7e3e

主页 - 搜索 Azure Active Directory - 应用注册 - 点击 拥有的应用程序 - 证书和密码 - 客户端密码 (客户提供)

主页 - 搜索 订阅 - 点击 订阅 - 订阅 ID (客户提供) [参考资料: 如何获取订阅权限](#)

百度云公有云

前提条件

- 支持的 api 版本:
 - v1

用到的百度云 SDK 中的 API

- list_zones - 获取可用区信息
- bcc.list_instances - 获取云服务器信息
- list_vpcs - 获取 VPC 信息
- list_subnets - 获取子网及网段信息
- list_enis - 获取云服务器的接口信息
- list_nats - 获取 NAT 网关信息
- get_route - 获取路由器信息
- list_peerconns - 获取对等连接信息
- list_security_groups - 获取安全组信息
- describe_loadbalancers - 获取负载均衡器信息
- rds.list_instances - 获取 RDS 信息
- list_eips - 获取外网 IP
- list_clusters - 获取容器 CCE 集群信息

举例

配置项	填写内容	备注
云平台名称	例: baidu-bce-1	云平台的名称
所属区域	在候选框中选择区域, 非必填	选择区域后表示云平台属于这个区域, 否则属于系统默认区域
资源同步控制器	在候选框中选择资源同步控制器, 非必填	选择资源同步控制器表示云平台在这个控制器上同步资源, 否则会自动选择所属区域的主节点同步资源; 如果

Access Key	例：AKIDztZ0C9dHuIQJwKMeZEixykjTBhz2L	没有指定所属区域则会自动选择从主控制器同步资源 在百度云页面 访问控制台-安全认证-Access Key 页面上获取用于 API 访问的 Access Key（客户提供）
Secret Key	例：itsHzkPo22jbtNZ61QEz3gc5bsPnXP	在百度云页面 访问控制台-安全认证-Access Key 页面上获取用于 API 访问的 Secret Key（客户提供）
API Endpoint	例：bj.baidubce.com	对接区域的 API Endpoint（服务域名）信息，不同区域会对应不同的 Endpoint。可参考 百度 BCC 产品文档 ，但注意去掉域名中的 bcc. 前缀。

限制

- 区域
 - 仅支持对接一个区域（百度云 SDK 没有获取区域的 API）
- NAT 网关
 - NAT 规则不支持同步（百度云 SDK 只能遍历 NAT 网关分别查询规则，无法统一获取）
- 负载均衡器
 - 负载均衡规则不支持同步（百度云 SDK 只能遍历负载均衡器分别查询规则，无法统一获取）
- 安全组
 - 安全组与虚拟机关联关系不支持同步（百度云 SDK 只能遍历虚拟机分别查询关联的安全组，无法统一获取）
 - 本地为安全组的入站规则不支持同步
 - 远端为安全组的出站规则不支持同步

容器

Kubernetes

前提条件

- 支持的 K8s 版本
 - Kubernetes v1.12
 - Kubernetes v1.13

- Kubernetes v1.16
- Kubernetes v1.18
- 支持的 K8s 发行版
 - OpenShift v1.11
 - 腾讯云 TKE
 - 阿里云 ACK
 - 青云 QKE/KubeSphere
- 支持的 CNI
 - Flannel
 - Calico
 - Multus
 - Open vSwitch
 - Weave [参考资料](#)
 - IPVlan [参考资料](#)
 - 腾讯云 TKE: [GlobalRouter](#)、[VPC-CNI 共享网卡](#)
 - 阿里云 ACK: [Terway](#)
 - 青云 QKE: [HostNIC](#)

举例

配置

项	填写内容	备注
云平台名称	例: kubernetes-1	云平台的名称
所属区域	在候选框中选择区域, 非必填	选择区域后表示云平台属于这个区域, 否则属于系统默认区域
资源同步控制器	在候选框中选择资源同步控制器, 非必填	选择资源同步控制器表示云平台在这个控制器上同步资源, 否则会自动选择所属区域的主节点同步资源; 如果没有指定所属区域则会自动选择从主控制器同步资源
额外对接路由接口	匹配接口名称的正则表达式, 格式参考页面默认值	作用是将节点上的其他功能性 IP 同步到 DeepFlow 中

建议: 额外对接路由接口描述了额外需要同步到 DeepFlow 中的接口 MAC 和 IP 地址。容器节点上通常会存在用于路由的接口, 例如: - Calico 用于 IP 隧道的 tunl0 - Calico 用于 VXLAN 隧道的 vxlan.calico - Flannel 用于路由的 cni0 - Flannel 用于隧道的 flannel.1

这些接口的 MAC 和 IP 地址均会出现在业务通信流中，因此需要将他们同步进来。若 CNI 还有除此之外的接口用于类似作用，需要对该正则表达式进行修改。但是需要注意的是，用于服务的 kube-ipvs 接口上的虚地址是不需要同步的，他们作用于全局，而非某个单一的容器节点，会通过 K8s SDK 自动同步。

Trident 配置项

- 参见容器采集器部署

其他

采集器同步-KVM

前提条件

- 对接限制
 - 不支持和其他云平台对接混合使用
 - 仅支持对接宿主机、虚拟机、MAC、IP 信息

对于 DeepFlow 未部署过的平台，实施之前挑选 3 个计算节点执行如下脚本，进行环境信息调研：

```
#####!/bin/sh
echo "===== VM List ====="
sudo virsh list
echo
echo "===== Interface List ====="
sudo ip a
echo
echo "===== VLAN config ====="
sudo cat /proc/net/vlan/config
echo
echo "===== Linux Bridge List ====="
sudo brctl show
echo
echo "===== OvS Show ====="
sudo ovs-vsctl show
echo
echo "===== OvS Port List ====="
sudo ovs-vsctl list port
echo
echo "===== OvS Interface List ====="
sudo ovs-vsctl list interface
echo
echo "===== VM Details ====="
echo
sudo virsh list | grep -E "^ *[0-9]+ " | awk '{print $1}' | while read
vid; do
    echo "===== VM" $vid "===== "
    sudo virsh dumpxml $vid
```

```
echo
sudo virsh domiflist $vid
done
```

如果 trident 已经启动并开始向控制器同步信息，可以在控制器上用下列命令进行信息查询：

```
mt genesis.trident-dump-platform-data vtap_ctrl_ip=$ctrl_ip
```

资源页面配置

配置项	填写内容	备注
云平台名称	例：采集器同步-1	云平台的名称，不可和其他云平台的名称重复。不支持中文名称，最大长度 64
所属区域	在候选框中选择区域，可不选	选择区域后表示云平台属于这个 region，如果不选表示学习此云平台上的区域
资源同步控制器	在候选框中选择资源同步控制器，可不选	选择资源同步控制器表示云平台在这个控制器上同步资源，不选会在区域的主节点上同步资源，如果没有区域会在主控制器上同步资源

系统页面配置

配置项	填写内容	备注
虚拟机 XML 文件夹	虚拟机 XML 文件位置	青云环境路径一般配置为/run/libvirt/qemu，其他环境一般用默认的/etc/libvirt/qemu

Trisolaris 配置

- 编辑主控制器上/usr/local/deepflow/saltstack/pillar/trisolaris.sls 中的如下参数，并执行以下命令行：

```
trident_type_for_unkonw_vtap: 1

salt -N controller saltutil.refresh_pillar
salt -N controller state.sls deepflow_config.trisolaris
salt -N proxy_controller state.sls deepflow_config.trisolaris
```

字段	含义
trident_type_for_unkonw_vtap	填写为 1（KVM 类型）

Genesis 配置

```
aging_time: 1d
default_region: default_region
default_vpc: default_vpc
host_ips:
- 172.20.1.136
- 172.20.1.132
```

```
- 172.20.1.106
local_ip_ranges:
- 10.0.0.0/8
- 172.16.0.0/12
- 192.168.0.0/16
- 169.254.0.0/15
- 224.0.0.0-240.255.255.255
```

字段	含义
aging_time	genesis 服务中数据的过期时间，可选值为 1d（一天）、1h（一小时）、1m（一分钟），如果过期时间小于一天就会导致丢数据，1h 和 1m 只能测试使用
default_region	默认的区域名称。所有的数据都会放到这个区域里面，开始学习数据后不可修改
default_vpc	默认的项目名称。如果某些资源没有项目，则会放入这个项目，开始学习数据后不可修改
host_ips	要对接的物理主机 ipa。即使没有这个机器，只要在这个配置项的 ip 就会被学习到，所以 host 信息不会被上述 platform_enabled 开关控制
local_ip_ranges	内网 IP 网段范围配置，配置的 IP 所在网络会被作为内网进行学习

支持单 IP，网段 CIDR
和 IP 范围三种配置方法

注意：只要网络内出现了一个外网 IP，就会被当作外网

- 采集器列表页面注册 genesis 所在主机的采集器（确保主机已经学习到）
- genesis 整个对接的过程如下
 - trident 收集云平台的资源，如 host, tenant, vm, network, port, ip 等，发送给 genesis 服务
 - genesis 服务把 trident 收集的资源进行整理通过 `http://127.0.0.1:20015/v1/info` 传递给 cloud-agent 服务
 - cloud-agent 服务通过 `http://127.0.0.1:20230/v1/info` 传递给 exchange 服务
 - exchange 服务把 trident 收集的资源存入数据库

采集器同步-Workload

前提条件

- 对接限制

- 不支持和其他云平台对接混合使用
- 仅支持对接云服务器、mac、IP 信息

页面配置举例

配置项	填写内容	备注
云平台名称	例：采集器同步-1	云平台的名称，不可和其他云平台的名称重复。不支持中文名称，最大长度 64
所属区域	在候选框中选择区域，可不选	选择区域后表示云平台属于这个 region，如果不选表示学习此云平台上的区域
资源同步控制器	在候选框中选择资源同步控制器，可不选	选择资源同步控制器表示云平台在这个控制器上同步资源，不选会在区域的主节点上同步资源，如果没有区域会在主控制器上同步资源

Trisolaris 配置

- 编辑主控制器上/usr/local/deepflow/saltstack/pillar/trisolaris.sls 中的如下参数，并执行以下命令行：

```
trident_type_for_unkonw_vtap: 3

salt -N controller saltutil.refresh_pillar
salt -N controller state.sls deepflow_config.trisolaris
salt -N proxy_controller state.sls deepflow_config.trisolaris
```

字段	含义
trident_type_for_unkonw_vtap	填写为 3（Workload-V 类型）

Genesis 配置

```
aging_time: 1d
default_region: default_region
default_vpc: default_vpc
local_ip_ranges:
- 10.0.0.0/8
- 172.16.0.0/12
- 192.168.0.0/16
- 169.254.0.0/15
- 224.0.0.0-240.255.255.255
```

字段	含义
aging_time	genesis 服务中数据的过期时间，可选值为 1d（一天）、1h（一小时）、1m（一分钟），如果过期时间小于一天就会导致丢数据，1h 和 1m 只能测试使用
default_region	默认的区域名称。所有的数据都会放到这个区域里面，开始学习数据后不可修改

default_vpc	默认的项目名称。如果某些资源没有项目，则会放入这个项目，开始学习数据后不可修改
local_ip_ranges	内网 IP 网段范围配置，配置的 IP 所在网络会被作为内网进行学习

支持单 IP，网段 CIDR 和 IP 范围三种配置方法

注意：只要网络内出现了一个外网 IP，就会被当作外网

- 采集器列表页面注册 genesis 所在主机的采集器（确保主机已经学习到）
- genesis 整个对接的过程如下
 - trident 收集云平台的资源，如 host, tenant, vm, network, port, ip 等，发送给 genesis 服务
 - genesis 服务把 trident 收集的资源进行整理通过 `http://127.0.0.1:20015/v1/info` 传递给 cloud-agent 服务
 - cloud-agent 服务通过 `http://127.0.0.1:20230/v1/info` 传递给 exchange 服务
 - exchange 服务把 trident 收集的资源存入数据库

配置文件录入

前提条件

- 5.5.3 版本开始支持

编辑 `/etc/cloud-agent.yaml`

```
filereader:  
- name: 'file #1'  
  enabled: true  
  path: '/etc/manual_data_samples.yaml'  
  format: 'yaml'
```

字段	含义
name	同上，如果名称和 Openstack 或 vSphere 配置的 name 一样，则把硬件设备合入 OpenStack 或 vSphere（数据合入目前只支持上述三个云平台），否则作为一个域单独学习
enabled	表示此块配置是否开启，值为 true 或 false
path	yaml 文件的位置
format	目前只支持 yaml 文件格式，值为 yaml
	当编辑 yaml 文件向云平台合入数据时，region、host、tenant、network、subnet 五种资源是支持去重的（host 按 ip 去重，其他四种按 name 去重），其他资源是直接进行合并

- 如果网络类型是 wan，且 external 为 true，则这个网络不学习，其下面的资源都会属于数据库初始化时的 isp 网络
- manual_data_samples.yaml 文件内容及字段说明，注意：文件需要放在控制器上，同时挂载到 cloud-agent 所在 Pod 上。

```
regions:
- name: region1
```

字段	含义
name	区域的名称，全局不可重复。类似从公有云平台获取 Regions

```
azs:
- name: az-1
  region: region1
```

字段	含义
name	可用区的名称，全局不可重复
region	可用区所属区域的名称

```
hosts:
- ip: 10.30.1.100
  memory_mb: 65536
  vcpus: 10
  htype: kvm
  az: az-1
  region: region1
```

字段	含义
ip	宿主机的 IP，不可重复
memory_mb	宿主机的内存大小，单位 MB
vcpus	宿主机的虚拟处理器个数
htype	宿主机虚拟化类型，可选值：kvm，hyper-v，默认 kvm
az	宿主机所属可用区的名称
region	宿主机所属区域的名称

```
tenants:
- name: tenant1
  region: region1
```

字段	含义
name	VPC 的名称，不可重复
region	VPC 所属区域的名称

```
users:
- name: user1
  email: user1@yunshan.net.cn
```

字段 含义

name 租户的名称, 不可重复

email 租户的邮箱

tenant_users:

- tenant_name: tenant1
user_name: user1

字段 含义

tenant_name VPC 的名称

user_name 租户的名称

networks:

- name: network1
external: false
net_type: wan
shared: true
tenant_name: tenant1
segmentation_id: 0
region: region1

字段 含义

name 网络的名称, 不可重复

external 可选值为 true/false。true 表示不学习此网络, 此网络下的资源会统一学习到 PublicNetwork; false 表示学习

net_type 可选值为 wan/lan。wan 表示外部网络, lan 表示内部网络

shared 可选值为 true/false。表示是否是共享网络

tenant_name 网络所属 VPC 的名称

segmentation_id VLAN 的 ID, 不知道可填 0

region 网络所属区域的名称

subnets:

- name: subnet1
network_name: network1
cidr: 10.30.0.0/16
gateway_ip: 10.30.0.1

字段 含义

name 子网的名称, 不可重复

network_name 子网所属网络的名称

cidr 子网的网段

gateway_ip 子网的网关 IP

routers:

- name: router1

```

tenant_name: tenant1
region: region1
ports:
- ip_address: '10.30.0.33'
  subnet_name: subnet1
  mac_address: '00:50:56:98:59:6f'

```

字段	含义
name	路由器的名称，不可重复
tenant_name	路由器所属 VPC 的名称
region	路由器所属区域的名称
ip_address	路由器上的 IP
subnet_name	路由器上 IP 所属子网的名称
mac_address	路由器上 IP 所对应的 MAC 地址，全局不可重复

```

vms:
- name: vm1
  tenant_name: tenant1
  launch_server: '10.30.1.100'
  az: az-1
  region: region1
  ports:
  - ip_address: '10.30.0.50'
    subnet_name: subnet1
    mac_address: '00:50:56:98:59:6e'

```

字段	含义
name	虚机的名称，不可重复
tenant_name	虚机所属 VPC 的名称
az	虚机所属可用区的名称
region	虚机所属区域的名称
ip_address	虚机上的 IP
subnet_name	虚机上 IP 所属子网的名称
mac_address	虚机上 IP 所对应的 MAC 地址，全局不可重复

F5

前提条件

- 支持的版本：
 - Kubernetes V14

部署

- 编辑 /etc/cloud-agent.yaml

```
f5:
- name: 'f5 #1'
  enabled: false
  region_uuid:
  host: '10.30.44.113'
  port: '443'
  user: 'user1'
  password: 'password1'
```

字段	含义
name	此块配置的名称（默认去掉所有空格），如果有两个相同类型平台对接，可填写两份 name 及以下的配置项，name 在这个文件中具有唯一性，不可和其他云平台配置模块的 name 重复。不支持中文名称，最大长度 64
enabled	表示此块配置是否开启
region_uuid	regions 全局配置项里面的 uuid，表示此平台属于这个 region，如果已经对接完成后需要修改此配置项，需要重新对接或参考升级后操作文档执行相应脚本
host	f5 提供服务的 ip
port	f5 提供服务的端口，一般为 443
user	f5 的登陆用户名，用于 cloud-agnet 服务获取数据
password	f5 的登陆密码，用于 cloud-agnet 服务获取数据

用到的 f5 SDK 模块

- f5.bigip.ManagementRoot
- f5.bigip.tm.auth.partitions
- f5.bigip.tm.ltm.virtuals
- f5.bigip.tm.ltm.pools
- f5.bigip.tm.ltm.snatpools
- f5.bigip.tm.ltm.snat_translations

调试工具

云平台 API 输出

配置开启 debug 模式

- 修改/etc/cloud-agent.yaml

debug: true
- 重启 cloud-agent systemctl restart cloud-agent

获取云平台原生 API 返回结果

- 执行 MT 命令 `mt platform_raw_data.list`

```
# resource_type 取值: 目录`/var/log/cloud-agent-debug/<domain_type>/`  
# <domain_name>`下的文件夹名称`  
mt platform_raw_data.list domain_type=<云平台类型> domain_name=<云平台名称> resource_type=<云平台 API/SDK 的资源类型>
```

调试容器资源同步

Kubernetes 的 Pod 学习不到 IP，排查步骤如下： - 控制器执行 `mt genesis.vinterfaces`，确认输出中是否有学不到的 IP - 如果没有，在控制器执行 `mt genesis.trident-stats`，可以查看 trident 的最后同步时间、IP 地址、类型和版本号等信息 - 可以检查没同步到 IP 的 Pod 所在 node 上的 trident 是否失联 - 如果目标采集器没有同步信息，检查采集器是否正确部署和正常运行，检查日志有无报错 - 如果目标采集器有同步信息，检查采集器日志有无报错 - 可能是容器权限问题 - 用 `mt genesis.trident-dump-platform-data vtap_ctrl_ip=<ip>`来检查同步的信息是否符合预期

扩容

简介

本章节描述 Deepflow 节点扩容方法。

扩容基本方法

- 增加 K8s node 节点，参考 K8s 部署篇
- 给需要扩容的角色打上对应 label

deployment 类型 Pod 会自动调度，daemonset 类型 Pod 在 label 打上后会启动

组件配置文件

简介

本章节描述 Deepflow 进程配置文件内容描述及配置下发方法。

配置修改原则

- 优先修改相关进程的 pillar 文件
- 若 pillar 文件中无所需修改的配置项再去手动修改对应 yaml 文件

配置下发方法

- pillar 文件修改后，执行 `mt deepflow.reconfig` 命令进行配置下发

```
# 该命令会对指定的角色进行配置下发和进程的重启，若未指定角色则会重启所有进程
```

```
mt deepflow.reconfig
# 可选参数 roles, 可以针对指定 roles 进行配置下发和重启进程
mt deepflow.reconfig roles=controller,tsdb
```

pillar 文件和 role 对应关系

所在路径为/user/local/deepflow/saltstack/pillar/:

路径	相关角色
bild.sls	trident,dedicated-trident
cloud-agent.sls	controller
dedicated_trident.sls	dedicated-trident
disk.sls	tsdb,elasticsearch-warm,dedicated-trident,controller
droplet.sls	tsdb
elasticsearch.sls	elasticsearch-warm
influxdb.sls	tsdb,controller
manager.sls	controller
mysql.sls	controller
pcap.sls	tsdb
proxy_controller.sls	proxy_controller
redis.sls	controller
trident.sls	trident
trisolaris.sls	controller,proxy_controller
tsdb.sls	tsdb

Trisolaris

配置文件

- 修改方法:

修改主控制器上/usr/local/deepflow/saltstack/pillar/trisolaris.sls 中的参数，并执行以下命令行：

```
salt -N controller saltutil.refresh_pillar
salt -N controller state.sls deepflow_config.trisolaris
salt -N proxy_controller state.sls deepflow_config.trisolaris
```

- trisolaris 运行配置:

```
# grpc 线程数目配置
grpc_workers_num:
# mysql 相关配置
mysql:
    database: deepflow
```

```
    user_name: root
    user_password: xxx
    host:
      - 127.0.0.1
    port: 20130
manager:
  host: 127.0.0.1
  port: 20403
warrant:
  host: 127.0.0.1
  port: 20413
# 日志级别
log_level: info
# api 监听端口
listen_port: 20014
# grpc 监听端口
trident_port: 20035
# ssl 监听端口
trident_ssl_port: 20135
# ssl 链接使用数据
ssl_cert_file: '/etc/pki/tls/certs/cert.csr'
ssl_key_file: '/etc/pki/tls/private/private.pem'
# 状态上报端口
statsd_port: 20040
# 调用其他进程超时时间
remote_api_timeout: 30
```

- trident 相关配置项:

```
# 通过采集器实现资源信息同步时使用
trident_type_for_unkonw_vtap: 0

# trident 升级版本号
trident_revision:

# trident 远程升级方式: http/grpc
# http 方式
self_update_url: http://172.16.1.105:23333/yum/trident
# grpc 方式
self_update_url: grpc

# trident path, 用于 grpc 方式远程升级
trident_linux_path: '/usr/local/deepflow/yum/trident'
trident_windows_path: '/usr/local/deepflow/yum/trident.exe'
```


Alarm

/etc/alarm.yaml 相关参数说明

告警使用全景图API 获取监控数据，底层数据监控存储存在一定的延迟，每次获取数据使用当前事件 delay 前的数据已保证数据已存入数据库

delay: 180

trigger 进程的数量，当报警策略增多时，将每条策略随机分配给 1 个 trigger 进程，避免 trigger 进程无法及时对每个策略进行触发判断

trigger_process_number: 3

告警事件保留时长

es_data_retention_time: 30 * 24 * 60 * 60 # 单位：秒，默认 30 天

ZooKeeper

/var/lib/zookeeper/conf/zoo.conf

功能: ZooKeeper 基本参数配置

- 配置：

```
# 心跳时间，单位是毫秒
tickTime=2000
# ZooKeeper 数据的存放位置
dataDir=/var/lib/zookeeper/data/
# 客户端监听端口，用于接收客户端请求
clientPort=2181
# 节点初始化时允许的最大 tick 数
# 时间的计算方法是 initLimit * tickTime
initLimit=5
# 节点间通信允许的最大 tick 数
syncLimit=2
```
- 操作: systemctl restart zookeeper

功能: ZooKeeper 节点配置

- 配置：

```
# server.x 中的 x 是节点 id，需要与 myid 文件中的内容相一致
# dfi-151 的部分是 hostname，不能用 ip 地址
# 2888 是节点之间进行通信的端口，3888 是选举 leader 的端口
server.1=dfi-151:2888:3888
server.2=deepflow-128:2888:3888
server.3=dfi-152:2888:3888
```
- 操作: systemctl restart zookeeper

`/var/lib/zookeeper/data/myid`

功能: ZooKeeper 的 id 文件

- 配置: 内容是一个数字, 与 zoo.cfg 中对应节点的节点 id 相一致
- 操作: `systemctl restart zookeeper`

Elasticsearch

配置文件

`elasticsearch.yml`

- 功能: 配置 elasticsearch 运行参数
- 内存配置:
 - `indices.memory.index_buffer_size`: 用于索引创建的内存量, 可参考: <https://www.elastic.co/guide/en/elasticsearch/guide/master/translog.html>
 - `thread_pool.bulk.queue_size`: 用于批量操作的队列大小, 设置过大会消耗更多内存
 - `indices fielddata.cache.size`: 用于倒排索引 key 的内存缓存大小, 超过此值时倒排索引 key 开始 swap 到文件, 可参考: https://www.elastic.co/guide/en/elasticsearch/guide/master/_limiting_memory_usage.html
 - `indices.breaker.total.limit`: 内存保护 (断路器) 的总阈值, 超过此阈值时请求返回失败, 可参考: <https://www.elastic.co/guide/en/elasticsearch/reference/current/circuit-breaker.html>
 - `indices.breaker.fielddata.limit`: 倒排索引 key 所用内存的最大值, 需要大于 `indices.fielddata.cache.size`
 - `indices.breaker.request.limit`: 每个请求用于排序、聚合等操作所消耗内存的最大值
 - `network.breaker.inflight_requests.limit`: 所有活跃请求的请求体 (request body) 所用内存的最大值
- 其他配置:
 - `cluster.name`: 集群名称, 所有节点应一致
 - `node.name`: 节点名称, 所有节点不能重复, 一般使用 `hostname`
 - `node.master`: 是否为 ES 集群主节点, 主节点控制整个集群, 一般选择 `zookeeper` 节点为主节点
 - `node.data`: 是否为 ES 集群数据节点, 数据节点保存数据并对数据执行 CRUD 操作
 - `node.attr.box_type`: `hot` 或 `warm` 节点

- hot 节点用于建立索引并存储请求较频繁的数据，索引是 IO 密集型操作，因此期望使用 SSD 作为存储设备
- warm 节点用于存储请求频繁度较低的数据，可以使用机械磁盘以提高 ES 集群存储容量
- node.data 为 false 时无需设置此属性
- path.data: ES 集群数据存储路径
- network.host: ES 的监听地址，一般为 0.0.0.0
- network.publish_host: ES 通告其他节点时所使用的地址，不能使用 0.0.0.0
- http.port: ES 的 HTTP 服务监听端口，用于 HTTP API 调用
- transport.tcp.port: ES 通过 TCP 接收数据的端口，用于 Native API 调用
- discovery.zen.ping.unicast.hosts: 用于节点发现的节点，通常包含所有的 master 节点
- discovery.zen.minimum_master_nodes: 设置为主节点数量 / 2 + 1
- 操作：
 - systemctl restart elasticsearch

elasticsearch.service

- 功能：配置 elasticsearch systemd 描述文件
- 配置：
 - LimitNOFILE=65536 放开 fd 文件描述符数量限制到 65536
 - LimitMEMLOCK=infinity 放开锁定内存限制，所谓锁定内存即内存页不会被移动到交换空间
- 操作：systemctl daemon-reload && systemctl restart elasticsearch

jvm.options

- 功能：配置 elasticsearch jvm 虚拟机运行参数
- 配置：
 - -Xms30g 设定初始内存占用至 30GB
 - -Xmx30g 设定内存最大占用至 30GB
- 操作：systemctl restart elasticsearch

单节点多 Elasticsearch 实例

诉求

JVM 的堆大小在超过 32GB 时，性能会急剧下降。在生产环境中往往有内存为 256GB 的服务器，如果单节点只运行一个 Elasticsearch 实例，能利用的内存有限。所以需要考虑在节点上运行多个 ES 实例，打破 32GB 的堆内存限制。

实现方法

Elasticsearch 可以支持单节点多实例，但不是天然支持的。在正常安装的 DeepFlow 分析节点中，配置步骤如下：

- 假定：ES 使用的磁盘为/mnt/sdb1、/mnt/sdc1 和/mnt/sdd1，希望在节点上运行三个 ES 实例。

- 停止并禁用原来的 Elasticsearch

```
systemctl stop elasticsearch.service
systemctl mask elasticsearch.service
```

- 创建实例用的配置文件夹，得到/etc/elasticsearch/{0,1,2}三个目录，分别用于三个实例的配置

```
cd /etc/elasticsearch
mkdir 0
mv * 0
cp -r 0 1
cp -r 0 2
chown -R root:elasticsearch /etc/elasticsearch
```

- 创建日志文件夹和 PID 文件夹，得到/var/log/elasticsearch/{0,1,2}和/var/run/elasticsearch/{0,1,2}

```
cd /var/log/elasticsearch
mkdir 0 1 2
chown elasticsearch:elasticsearch 0 1 2
cd /var/run/elasticsearch
mkdir 0 1 2
chown elasticsearch:elasticsearch 0 1 2
```

- 修改三个实例的配置文件，/etc/elasticsearch/{0,1,2}/elasticsearch.yml 的以下项需要修改
 - **node.name:** 在原来名字的后面添加实例号，如原来叫 dfi-151 的话，新的名字分别是 dfi-151-0、dfi-151-1 和 dfi-151-2
 - **node.master:** 如果原来的节点是 true，实例 0 不需要修改，实例 1 和实例 2 需要改为 false；如果原来是 false 则都不需要修改
 - **path.data:** 三个实例都需要修改，将原来的磁盘均摊开。本例中，实例 0 用/mnt/sdb1，实例 1 用/mnt/sdc1，以此类推
 - **http.port:** 实例 0 不需要修改，实例 1 改为 20062，实例 2 改为 20064，以此类推，可以占用 20061-20070 中的偶数端口

- `transport.port`: 实例 0 不需要修改, 实例 1 改为 20061, 实例 2 改为 20063, 以此类推, 可以占用 20061-20070 中的奇数端口
- `discovery.zen.ping.unicast.hosts`: 实例 0 不需要修改, 其余实例均需要在列表中每一个 `hostname` 后面加上端口号 20041, 并且两端加上引号。

如

```
discovery.zen.ping.unicast.hosts: [dfi-151, deepflow-128, dfi-152, deepflow-154]
```

改为

```
discovery.zen.ping.unicast.hosts: ["dfi-151:20041", "deepflow-128:20041", "dfi-152:20041", "deepflow-154:20041"]
```

- 添加 `shard allocation` 相关字段, `/etc/elasticsearch/{0,1,2}/jvm.options` 的以下项需要修改

```
# 本机的 hostname
node.attr.host: dfi-151
cluster.routing.allocation.awareness.attributes: host
# 所有 data 节点的 hostname
cluster.routing.allocation.awareness.force.host.values: dfi-151,dfi-152,dfi-153,dfi-171
```

- `-Xmx30g` 设定内存最大占用至 30GB, 改为适当的值
- `-Xms30g` 设定初始内存占用至 30GB, 改为与 `-Xmx` 一致

- 修改 `systemd` 脚本

```
mv /lib/systemd/system/elasticsearch.service /lib/systemd/system/elasticsearch@.service
```

将上述文件的以下几个参数做如下修改

```
[Unit]
Description=Elasticsearch %I
```

```
[Service]
Environment=CONF_DIR=/etc/elasticsearch/%i
Environment=LOG_DIR=/var/log/elasticsearch/%i
Environment=PID_DIR=/var/run/elasticsearch/%i
```

添加

```
[Service]
Environment=ES_JVM_OPTIONS=/etc/elasticsearch/%i/jvm.options
```

- 清理转移到新 Elasticsearch 实例上磁盘里的内容。ES 数据目录下的 `_state/node-xxx.st` 文件记录了原节点的信息，如果不删除，集群会认为新增加的实例和原来的实例是同一个节点。可以选择以下两个方法之一解决这个问题
 - 删除整个 `/mnt/sd*1/es` 目录里的内容，数据需要重新进行同步
 - 修改 `node-xxx.st` 文件，写入一个新的 node ID。可能会减少需要同步的数据量，不过没有详细验证
- 启动 Elasticsearch


```
systemctl start elasticsearch@0
systemctl start elasticsearch@1
systemctl start elasticsearch@2
```
- 添加 Elasticsearch 自启动


```
systemctl enable elasticsearch@0
systemctl enable elasticsearch@1
systemctl enable elasticsearch@2
```
- 检查 Elasticsearch 状态
 - 从原有实例查看新实例是否能查到


```
curl 127.0.0.1:20042/_cat/nodes
```
 - 从新实例查看原有实例是否能查到


```
curl 127.0.0.1:20062/_cat/nodes
curl 127.0.0.1:20064/_cat/nodes
```
 - 从新实例做一些简单的查询


```
curl 127.0.0.1:20062/dfi_flow__0_*/_search
```

参考资料

- <https://www.elastic.co/guide/en/elasticsearch/reference/current/configuring-tls.html>
- <https://www.elastic.co/guide/en/elasticsearch/reference/current/get-started-enable-security.html>
- <https://www.elastic.co/guide/en/elasticsearch/reference/current/http-clients.html>

开源组件密码配置

MySQL 修改密码

`mysqladmin -u 用户名 -p 旧密码 password 新密码`

Redis 修改密码

- `/etc/redis.conf`
`requirepass [密码]`
- `systemctl restart redis`

开启 Elasticsearch 认证

操作步骤

- 生成证书。在某一个 es 节点执行

```
/usr/share/elasticsearch/bin/elasticsearch-certutil ca  
/usr/share/elasticsearch/bin/elasticsearch-certutil cert --ca elastic-stack-ca.p12
```

- 将生成的 `elastic-certificates.p12` 文件放置到每一个 elasticsearch 节点的 `/etc/elasticsearch/` 目录下，权限改为 `elasticsearch:elasticsearch`
- 修改每个 elasticsearch 节点的 `/etc/elasticsearch/elasticsearch.yml`，添加如下配置

```
xpack.graph.enabled: false  
xpack.ml.enabled: false  
xpack.monitoring.enabled: false  
xpack.security.enabled: true  
xpack.security.transport.ssl.enabled: true  
xpack.security.transport.ssl.verification_mode: certificate  
xpack.security.transport.ssl.keystore.path: elastic-certificates.p12  
xpack.security.transport.ssl.truststore.path: elastic-certificates.p12  
xpack.watcher.enabled: false
```

- 重启所有 elasticsearch
- 初始化密码（比如都设置成 `security421`）。在某一个 es 节点执行

```
/usr/share/elasticsearch/bin/elasticsearch-setup-passwords interactive
```

- 检查配置。在 es 节点执行下列命令正常显示 elasticsearch 的信息即可

```
curl --user elastic:security421 127.0.0.1:20042/
```

influxdb 修改密码

`SET PASSWORD FOR 用户 = '密码'`

zookeeper 修改密码

```
/var/lib/zookeeper/bin/zkCli.sh  
addauth digest heng:123456
```

deepflow 进程密码修改

- /etc/salt/master.d/pillar 文件夹内，找到对应开源组件名字的 sls
- 修改密码配置
- mt deepflow.reconfig

clickhouse 密码修改

- 修改/etc/salt/master.d/pillar/clickhouse.sls 中的配置

```
clickhouse:  
  username: user  
  password: pass
```

- 应用配置并重启 clickhouse

```
salt -N tsdb state.apply clickhouse.config  
salt -N tsdb state.apply clickhouse.stop  
salt -N tsdb state.apply clickhouse.start
```

参考资料

- <https://www.elastic.co/guide/en/elasticsearch/reference/current/configuring-tls.html>
- <https://www.elastic.co/guide/en/elasticsearch/reference/current/get-started-enable-security.html>
- <https://www.elastic.co/guide/en/elasticsearch/reference/current/http-clients.html>

界面配置

web 界面基本参数配置

修改镜像配置文件/usr/local/deepflow/templates/values-custom.yaml 中 dfWebCore 一节 app_config_user.js 部分， 保存文件后执行 /usr/local/deepflow/bin/deepflow-deploy -uo df-web-core 用于覆盖默认配置

- 配置：

```
{  
  APPLICATION_LAYER_HIDE: false, // 全景图屏蔽应用层数据  
  VTAP_HIDE: false, // 甜橙控制采集器页面隐藏  
  TO_DASHBOARD_PATH: '/dashboard', // 指定顶部图标跳转路径, 默认应该是/  
  dashboard  
  SAAS_TENANT_TO_DASHBOARD_PATH: 'https://deepflow.yunshan.net', //  
  saas 环境下租户指定顶部链接
```



```
EXTERNAL_LINK: true, // 外部链接功能是否对外开发
API_TIMEOUT: 60 * 1000, // 请求API 超时时间
API_UPLOAD_TIMEOUT: 60 * 1000, // 请求上传相关API 超时时间
MINIMUM_BROWSER_VERSION: 60, // 浏览器最低版本
MAX_LABEL_NUM: 100,
MAX_LABEL_LENGTH: 32,
AES_CONFIG: { // aes 配置需与后端保持一致, 意义不大
  key: 'JeF8U9wHFOMfs2Y8',
  iv: '1234567890123456'
},
SM4_CONFIG: {
  key: 'JeF8U9wHFOMfs2Y8',
  mode: 'ecb',
  cipherType: 'base64'
},
FORCE_REPORT: true, // 回溯是否支持force_report 状态配置
HELP_VISIBLE: ['admin'], // 默认只有管理员可见右上角帮助链接, 改为['
admin', 'tenant']后租户也可见
ADMIN_TENANT_VISIBLE: true, // 默认管理员可见租户列表, 改为false 后
不可见
FLOW_VIEW_MULTIPLE_SELECT_LEN: 25, // 全景图多选下拉框最多可选数目
FLOW_VIEW_HISTORY_SEARCH_BTN_NUM: 8, // 全景图历史搜索快捷按钮数目
RESOURCE_AUTHORIZE_MULTIPLE_SELECT_LEN: 25, // 账户管理最多可给租户
授权数目
SELECT_DISPLAY_MAX_LEN: 100, // 下拉框最多展示条目数量
SUPPORT_SECOND_DATA_TIME: 3600 * 36, // 距当前时间小于等于该值时视
图支持秒级数据的选项, 时间单位为s
FLOW_VIEW_CUSTOM_LINE_CHART_NUM: 20, // 全景图支持自定义的折线数目
VIEW_UPDATE: false, // 视图升级按钮, 默认关闭
WINDOW_SIZE: 60, // 折线图滑窗下拉框选项上限
FLOW_VIEW_MAX_RESOURCE_SET: 10, // 全景图允许添加资源集的最大数目
VIEW_TABLE_SELECT_MULTIPLE_LIMIT: 40, // 流日志table 选择显示列下拉
框最多可以选择的项目数
SAFE_METHOD: false, // 开启安全方法过滤, 替换请求方法
DOMAIN_ATTACHED_K8S_COUNT: 10, // 云平台可添加k8sconfig 的个数
FLOW_VIEW_KNOWLEDGE_NODE_COUNT: 2, // 全景图知识图谱拓扑图每类节点的
展示个数
SAAS_DEFAULT_SEND: ['admin', 'tenant'], // SaaS 环境, 全景图管理员,
租户默认发起请求, 移除子项后对应权限不再默认发起请求
CHART_COLOR: [ // 图表颜色
  "#16a9e8",
  "#45b0d6",
  "#2470a0",
  "#003ca5",
  "#001e72",
  "#8498d1",
  "#bb76cc",
```

```

"#aa48bc",
"#8d1fa0",
"#7039c6",
"#9cd3c3",
"#49b292",
"#9ac965",
"#78b446",
"#56863a",
"#eadb92",
"#e5c333",
"#dba638",
"#d17859",
"#994d77"
],
CHART_ALARM_COLOR: 'rgba(255, 0, 0, 0.5)',
TOPO_COLORS: {
    LINK_WITHDATA_DEFAULT: '#4C536E', // 流量线默认颜色
    LINK_AUTOFILLED_DEFAULT: 'green', // 自动添补的关系连线颜色
    LINK_MANUALFILLED_DEFAULT: 'green', // 手动添补的关系连线颜色
    NODE_OUTLINE_DEFAULT: '#4C536E', // 节点外圈颜色
    LINK_PHYSICAL_HIGHLIGHT: 'green', // 物理拓扑连线高亮颜色
},
// 需要隐藏的菜单路径
HIDDEN_MENU_PATH_LIST: [],
// sentry 错误上报 url
// 空字符串: 不初始化 sentry
// 非空字符串: 初始化 sentry 并上报访问信息
SENTRY_DSN_URL: '',
// 忘记密码链接地址
FORGET_PASSWORD_LINK_URL: 'https://deepflow.yunshan.net/phone-code.html',
// 控制"未授权"标识的显隐
SHOW_UNAUTHORIZE_TAG: true,
// 表格 size 配置
// 可选值: 'mini', 'medium', 'large'
// 主要影响: 行高, 文字大小
// 作用范围: 应用, 网络下 列表
VIEW_TABLE_SIZE: 'mini',
// 表格 最大展示数量 配置
// 超出此数量, 展示滚动条
VIEW_TABLE_MAX_LEN: 100
}

```

manager

/etc/manager.yaml 相关参数说明

部署模式, 默认 npm, 可选 npb

deploy_type: npm 注意: 若部署 npb 此配置项必须改为 npb 不然无法正常工作

系统日志、操作日志、APM 监控数据保留时长

`es_data_retention_time: 30 * 24 * 60 * 60 # 单位: 秒, 默认 30 天`

采集器默认授权类型

指定授权类型仅对支持该类型的采集器生效

可选值: A/B/C/空, 为空时使用 A 类或 B 类授权

`original_authorization: C`

monitor

/etc/monitor.yaml 相关参数说明

资源变更事件数据保留时长

`es_data_retention_time: 30 * 24 * 60 * 60 # 单位: 秒, 默认 30 天`

bild

bild 部署

bild 功能

- 流量负载均衡, 基于 eBPF/XDP 技术, 将 N (N>=1) 个指定源接口 (src_tap_interfaces) 上接收到的 (ingress) /发送出去的 (egress) 流量, 通过对称哈希, 均摊到其他 M (M>=1) 个的指定目的接口 (dst_tap_interfaces) 的接收通道 (ingress) /发送通道 (egress), 支持 double VLAN, erspan 隧道解封装, 哈希支持 TCP/UDP/TCP6/UDP6 的五元组计算。
- 物理网卡性能优化。解决复杂的网络场景下, 对物理网卡特定优化使其达到更好的性能。包含:
 - 网卡 RSS 支持对称哈希
 - 网卡队列中断 CPU 绑定
 - 网卡队列数量设置
 - 环形 rx 描述符的大小设置
 - 设置流量类型(TCP, UDP 等)基于五元组的均衡分摊
 - 支持 XDP CPU Redirect 功能解决 double vlan/erspan/vxlan 类型数据无法均衡到不同的 CPU, 造成某一个 CPU 过载的情景。

网卡驱动升级

注意: 只针对于 Intel X722 10Gb, Intel X710 10Gb 网卡使用 i40e 驱动并且使用 XDP CPU Redirect 做优化的情况, 需要升级驱动到 version: 2.11.25。其他场景不需要升级。

- 安装

- 使用 `ethtool -i <接口名字>` 查看 i40e 版本，如果不是 2.11.25 则进行升级。
- 使用控制器 `/usr/local/deepflow/yum/7/` 下的 `i40e-2.11.25-1.x86_64.rpm` 进行安装升级，安装命令如下：

```
yum --disablerepo=* localinstall -y ./i40e-2.11.25-1.x86_64.rpm
```

- 生效

准备脚本：i40e_update.sh

```
#!/bin/bash
rmmod i40e
modprobe i40e
systemctl restart network
```

使用脚本进行实际替换驱动，注意：这里使用脚本是为了防止要升级的网卡作为管理接口，手动生效后 IP 会丢失使得 SSH 链接失败的情况，解决办法是使用 `nohup` 方式执行脚本。

```
chmod +x i40e_update.sh
nohup ./i40e_update.sh &
```

CPU 隔离

注意：只针对于物理网卡使用 XDP CPU Redirect 做优化的情况，其他场景不需要 CPU 隔离。

- 为了排除系统任务调度对物理网卡收包的干扰，需要隔离物理核心。
 - 隔离的 CPU 核心，要和物理网卡在同一个 NUMA 节点上。可以通过下面方法获得物理网卡所在的 NUMA 节点：

```
# 如：物理网卡为p1p2,先得到PCI 地址:
ls /sys/class/net/p1p2/subsystem/ -al | grep p1p2 | awk -F'/net' '{print $1}' | awk -F '/' '{print $NF}'
0000:41:00.1
lspci -vvv -s 0000:41:00.1 | grep "NUMA node:"
    NUMA node: 1
```

- 获得逻辑核和物理核的对应表

```
# 如得到NUMA node: 1 的所有逻辑核和物理核的对应表
for cpu in `ls /sys/bus/node/devices/node1/cpu*/topology/core_id`; do echo "$cpu <-> `cat $cpu`"; done
/sys/bus/node/devices/node1/cpu11/topology/core_id <-> 5
/sys/bus/node/devices/node1/cpu13/topology/core_id <-> 0
/sys/bus/node/devices/node1/cpu1/topology/core_id <-> 0
```

```
/sys/bus/node/devices/node1/cpu23/topology/core_id <-> 5
.....
```

隔离整个物理核,如隔离 core 5, 就需要把逻辑核 cpu11, cpu23 隔离即可。

- 根据经验值来判断隔离 core 的数量。

核心隔离数量和性能对照:

离物理核心数量	收包 PPS
2	1.11M
3	1.76M
4	2.51M

- 隔离操作

- 在文件 /boot/grub2/grub.cfg 中找到 menuentry 'CentOS Linux (4.19.17-1.el7.x86_64) 7 (Core)' 在 linux16 行末添加“isolcpus=要隔离的 CPU 列表”，例如:

```
linux16 /vmlinuz-4.19.17-1.el7.x86_64 root=/dev/mapper/centos
-root ro crashkernel=256M rd.lvm.lv=centos/root rd.lvm.lv=cen
tos/swap rhgb quiet spectre_v2=off nopti pti=off idle=poll LA
NG=en_US.UTF-8 iommu=pt isolcpus=11,23
```

添加 isolcpus=11,23, 其中的 11,23 即是 core 5 的两个逻辑核心。

- 注意: 需要重启机器才能使隔离生效。

部署和配置

注意: 注意: 部署要求 CentOS 7, linux 4.19 内核

salt 环境下部署

- 部署 deepflow 时, bild 默认处于 disable 状态。
- 如需 enable bild, 需修改 /usr/local/deepflow/saltstack/pillar/bild.sls 中的 bild 配置

```
bild:
  dedicated-trident:
    enable: False
    dst_interface_name_prefix: tap
    dst_interface_count: 8
  trident:
    enable: False
    dst_interface_name_prefix: tap
    dst_interface_count: 1
```

- bild 运行在 trident 和 dedicated-trident 节点
- enable: False
 - enable/disable bild 功能，默认 False，如果启用使用 True
- dst_tap_interface_name_prefix: tap
 - 创建虚拟接口的接口名前缀，默认 tap
- dst_tap_interface_count 8/1
 - 创建的虚拟接口数量，dedicated-trident 节点默认 8 个，trident 节点默认 1 个。
- 然后将 bild 部署到相应的节点

- 部署到 dedicated-trident 节点：

```
# 配置文件更新
salt-run state.orch dedicated_trident_config
# 关闭trident、bild
salt-run state.orch dedicated_trident_stop
# 启动trident、bild
salt-run state.orch dedicated_trident_start
```

- 部署到 trident 节点：

```
# 配置文件更新
salt-run state.orch trident_config
# 关闭trident、bild
salt-run state.orch trident_stop
# 启动trident、bild
salt-run state.orch trident_start
```

容器环境下部署

- 获取 bild 和 iproute 安装包
 - 从 ISO 中的 extra-artifaces/df-bild-package 获取。
- 安装

```
yum --disablerepo=* localinstall -y ./{iproute,bild}/*.rpm
```

说明：bild 对系统包 glibc 有一定的要求，需要 $\geq 2.17-260$ 如果版本过低需要通过 yum update glibc 进行更新。

- 创建虚接口
 - 在文件/sbin/ifup-local 追加下面，添加权限并执行 chmod +x /sbin/ifup-local; sh /sbin/ifup-local

```
for i in {0..7}
do
    /sbin/ip link add tap-bild-$i type dummy
```

```
    /sbin/ip link set tap-bild-$i promisc on
    /sbin/ip link set dev tap-bild-$i up
done
```

- 修改配置文件/etc/bild.conf(下面举例)

ebpf:

```
- src_tap_interfaces: em1
  src_direction: 0
  dst_tap_interfaces: tap-bild-0,tap-bild-1,tap-bild-2,tap-bild-3,
tap-bild-4,tap-bild-5,tap-bild-6,tap-bild-7
  dst_direction: 0
```

- 启动 bild

```
systemctl enable bild.service
systemctl start bild.service
```

- 修改 trident 采集接口的配置（修改成上面建立的 8 个虚拟接口）并重启 trident。

配置网卡优化功能

- 根据实际需求可以对网卡做对称哈希，XDP CPU 重定向等优化。根据实际情况可以对某个节点单独修改/etc/bild.conf 文件。

```
nic_optimize:
- interface: p1p2
  rss_channel_count: 1
  hi_cpu_list: 2
  xdp_cpu_redirect: true
  xdp_cpu_redirect_list: 5,6,7,8,9,10
```

以上配置是样例，如果要对网卡做优化需要增加上面配置项，具体配置含义参照 NIC 优化配置章节。

- 某个节点添加配置后，重启 bild。systemctl restart bild

基本操作

- 启动
 - systemctl start bild
- 停止
 - systemctl stop bild
- 查看状态
 - systemctl status bild
 - bild-ctl status
- 版本信息
 - bild-ctl version

配置文件/etc/bild.conf 说明

接口均摊配置

ebpf:

```
- src_tap_interfaces: lo
  src_direction: 0
  dst_tap_interfaces: lo
  dst_direction: 0
```

- src_tap_interfaces
 - 源接口
 - 最大支持 32 个接口
 - 接口直接用,分割, 且不能有空格、tab 等空白字符
- dst_tap_interfaces
 - 目的接口
 - 最大支持 128 个接口, 因为 trident 目前最多支持 16 个接口, 所以实际设置接口的数量不要超过 16。
 - 接口直接用,分割, 且不能有空格、tab 等空白字符
- src_direction
 - 源接口流量方向
 - 0: ingress, 1: egress
- dst_direction
 - 目的接口流量方向
 - 0: ingress, 1: egress

NIC 优化配置

nic_optimize:

```
- interface: p1p2
  rss_channel_count: 1
  hi_cpu_list: 2
  xdp_cpu_redirect: true
  xdp_cpu_redirect_list: 5,6,7,8,9,10
```

- 如果不进行网卡优化可注释掉上面配置选项, 默认: 不启用网卡优化, 处于注释状态。
- interface
 - 要优化配置接口的名称, 如: interface: p1p2
 - 这个接口是 src_tap_interfaces 中的一个, 不能写多个如: interface: p1p2,p2p1, 如果要进行多个接口做优化可以如下配置:

```
nic_optimize:
- interface: p1p2
  .....
```



```
- interface: p1p1
.....
```

- `rss_channel_count`
 - RSS 硬件队列数量，数据流在物理网卡上做哈希之后，均摊到多少个队列上并在此队列上发出中断，最大可配置 16 且不要超过逻辑 CPU 核数，当启用 XDP CPU Redirect 时配置为 1
- `hi_cpu_list`
 - 中断和 CPU 绑定，值：CPU-ID/CPU-ID 列表
 - 处理硬件中断的 CPU 列表或单个 CPU，数量最好等于 `rss_channel_count`，但流量较小时一个 CPU 可处理多个硬件中断队列，当启用 XDP CPU Redirect 时只需指定 1 个 CPU 即可。
 - 如果启用了 CPU 隔离，例如：`isolcpus=11,23,1,13` 并且启用了 XDP CPU Redirect 功能，那么需要在隔离的 CPU 列表中挑选一个，如设置成：`hi_cpu_list: 11`
 - 对于不启用 XDP CPU Redirect 功能情况配置：
 - 可使用 `local`，自动匹配本地 numa 上的 CPU。即：`hi_cpu_list: local`
 - 可使用 `all`，自动匹配所有 CPU。即：`hi_cpu_list: all`
 - 可指定 CPU 列表，如 `hi_cpu_list: 2,4,5,6,7,8`
 - 注意：指定的 CPU 和物理网卡在同一个 NUMA 上。物理网卡所在的 numa 节点以及 CPU 分布可以 `/usr/share/bird/scripts/bird-nic-check.sh <接口名字>` 看到，也可使用 `cpu_layout` 命令查看 CPU 在 NUMA 节点上的分布。
- `xdp_cpu_redirect`
 - 是否开启 XDP CPUMAP redirect 功能。值：`true/false`
 - `xdp_cpu_redirect` 针对 double VLAN，`erspan` 等数据包无法使物理网卡把数据分散到不同 CPU 进行处理，而是造成某一个 CPU 过载，出现大量丢包的场景进行优化。
- `xdp_cpu_redirect_list`
 - XDP CPU Redirect 后处理包的 CPU 列表
 - 需要考虑重定向的 CPU 所在的 numa 节点和物理网卡所在的 numa 节点保持一致。物理网卡所在的 numa 节点以及 CPU 分布可以 `/usr/share/bird/scripts/bird-nic-check.sh <接口名字>` 看到。
 - 注意：为了不让系统调度干扰，建议使用 `isolcpus`，隔离出整个物理核心，并且这些 CPU 和物理网卡在同一个 NUMA 上。例如：`isolcpus=11,23,1,13` 其中 11，可作为 RSS 接收 CPU，其他被 XDP 使用，因此可配置成 `xdp_cpu_redirect_list: 23,1,13`
 - 此项只有 `xdp_cpu_redirect: true` 才有意义。

注意： 下面四个配置项是可选配置项，若无特别需求可不用关心采用默认值即可。

- `symmetric_hash`
 - 是否使用 RSS 对称哈希。目的是避免应用层收到的数据包出现乱序情况。值: `true/false`, 默认是 `true`
- `hi_delay_usec`
 - 硬件中断延迟, 一般无需配置, 增大延迟可在 CPU 硬件中断过高时优化性能, 最大可配置 125
 - 默认配置成 `hi_delay_usec: 0`, 0 表示中断不做延迟马上让 CPU 收包, 让 CPU 处理更多的数据包。
- `xdp_qsize`
 - XDP 重定向 CPU 的队列大小设置, 默认配置为 `xdp_qsize: 2048`
 - 此项只有 `xdp_cpu_redirect: true` 才有意义。
- `ethtool_cmd`
 - 直接执行 `ethtool` 命令。
 - 默认值配置为 `ethtool_cmd ""`。
- 启用 `nic_optimize` 后 `/usr/share/bird/data/nic_info.<接口名字>` 会备份硬件网卡信息。
- 为了得到更好的性能, 会停用 `irqbalance` 服务。

注意: 注: 修改配置文件后, 需重启 `bird` 才能生效

调试方法

功能: 查看实际配置, 统计和调试信息

- 通过 `bird-map` 命令, 查看实际配置, 统计信息
 - 查看实际配置


```
bird-map lookup
```
 - 查看统计信息
 - 查看所有源接口, 每种数据方向上, 数据包被均摊到每个目的接口的统计信息


```
bird-map stats redirect
```
 - 查看指定源接口, 指定源/目的方向上, 数据包被均摊到每个目的接口的统计信息


```
# 查看源接口 p2p2 均摊数据包的统计
bird-map stats redirect p2p2 0 0
```
 - 查看指定目的接口, 指定方向上, 收到的数据统计信息

```
build-map stats rxtx tap0,tap1 0
# build-stats.sh 工具只统计ingress 方向。
/usr/share/build/scripts/build-stats.sh tap0,tap1,tap2,tap3,tap4,tap5,tap6,tap7
```

注意： 注： 打印信息中，均用 ifindex 代替接口名称； 0: ingress, 1: egress

- ebpf 调试

- 开启/关闭

```
# 开启
build-ctl debug
# 关闭
build-ctl nodebug
```

- 查看

```
tc exec bpf dbg
```

注意： ebpf 调试时，确保
/proc/sys/kernel/ftrace_enabled,/sys/kernel/debug/tracing/tracing_on 都为 1

- XDP CPU Redirect 调试，查看 CPU 重定向统计。

```
# build-ctl xdp-redirect <物理网卡接口名字>
# 例如查看物理网卡 p2p2 的详细CPU 重定向数据统计。
build-ctl xdp-redirect p2p2
```

- 查看启动/关闭日志信息

- 可在/var/log/messages 中查看 build 日志

日常运维操作

软件升级

本章节描述 DeepFlow 的版本升级方法。

升级前操作

v6.1.0

注意： 升级 v6.1.0 引入的不兼容问题

- 由于应用-HTTP/RPC/SQL/NoSQL/MQ/DNS 页面合并为统一的应用-流日志页面，历史数据不保留

- 由于应用-流日志页面快速搜索框的大幅升级，应用-流日志-详情页面中的搜索条件将不会在升级后保留
- 由于 eBPF 的加入以及为以后应用协议增加做准备，应用-服务/路径/流日志页面中应用协议的枚举值有变更，详见下表

应用协议类型	v6.0 及之前的枚举值	v6.1 及之后的枚举值
未知	0	0
其他	7	1
HTTP	1	20: HTTP1, 21: HTTP2
Dubbo	5	40
MySQL	3	60
Redis	4	80
Kafka	6	100
DNS	2	120

升级 v6.1.0 之前的其他操作

暂无

v6.0.0

准备升级至 v6.0.0 之前的操作

按顺序执行以下操作

（按需）升级 Patch

检查 Patch 是否完整，如未升级到最新 Patch，需将 5.7.3 包含数据库升级的 Patch 全部打上 - 13 号 Patch - 21 号 Patch

备份 MySQL 数据

```
##### 进入 mntnct 服务对应的 POD 执行 kubectl exec -it -n deepflow pod-xxxxxxx sh
mysqldump -hmysql -uroot -p -P30130 deepflow --hex-blob > /usr/local/deepflow/patch/deepflow_$(date +%Y%m%d).sql
```

备份原 values.yaml 配置

600 之后版本，values.yaml 仅做默认模板，如需修改自定义参数，请修改 values-custom.yaml。

```
##### 参考 diff 配置方法
##### 对比文件
diff /media/templates/values.yaml /usr/local/deepflow/templates/values.yaml
##### 只看变动
```

```
diff /media/templates/values.yaml /usr/local/deepflow/templates/values.yaml | grep ">"
```

卸载 5.7.3 版本

```
/usr/local/deepflow/bin/deepflow-deploy -e
```

删除 /etc/hosts 自定义域名

6.0.0 版本之后已取消写 hosts 操作，由 `opensource-endpoints-operator` 组件替代，需删除 5.7.3 版本中写入 hosts 的自定义域名

参考：

登录主从区域所有节点查看 /etc/hosts 是否存在类似配置，删除即可

```
10.1.20.21 master.mysql
10.1.20.21 master.manager
10.1.20.21 master.warrant
10.1.20.21 master.talker
```

chrony 时间同步配置

6.0.0 版本后需由 K8s 集群宿主直接提供，参考安装文档 FAQ 部分《如何启动 ntp 服务》，启动 master 节点 NTP 服务。

```
echo "allow all" >> /etc/chrony.conf
systemctl restart chronyd
```

v5.7.3

准备升级至 v5.7.3 之前的操作

备份 MySQL 数据

如本机没有 `mysql`，可使用 `mntnct` 或 `mysql pod` 执行

```
mysqldump -hmysql -uroot -p -P30130 deepflow --hex-blob > /usr/local/deepflow/patch/deepflow.sql
```

重命名原数据目录，防止干扰新数据

```
mv /var/lib/mysql /var/lib/mysql_$(date +%Y%m%d)_bak
```

备份其他数据目录

备份数据前，需先停止 Salt 部署的服务，参考下节《停止 Salt 模式服务》命令

需 OP 根据当前 Salt 部署环境，找到原数据目录，并通过备份迁移，或软连接的方式，挂载到容器化默认路径下 - 源路径查询参考方式: `cat /usr/local/deepflow/saltstack/pillar/disk.sls`

容器默认路径参考容器化部署文档 - 依赖 - 数据路径 - `pcap: /mnt/pcap - elasticsearch: /mnt/es - influxdb-data: /mnt/influxdb/data - influxdb-meta: /mnt/influxdb/meta - clickhouse-data: /mnt/clickhouse` # 如有多个源文件夹，选择 0-tiers 结尾的 - `clickhouse: /var/lib/clickhouse.0/` # 如有多个源文件夹，选择

0-tiers 结尾的，且需修改 templates/values.yaml 中 clickhouse 的 path:
/var/lib/clickhouse 为正确目录，示例：path: /var/lib/clickhouse.0

注意：如 clickhouse 或 elasticsearch Pod 启动报错，可能为数据目录权限不对，
参考如下修改目录权限 - clickhouse 目录使用 chown -R 101:101
/var/lib/clickhouse.0/ - clickhouse-data 目录使用 chown -R 101:101
/mnt/clickhouse - elasticsearch 目录使用 chown -R elasticsearch:elasticsearch
/mnt/es

停止 Salt 模式服务

```
systemctl disable --now grafana-server
systemctl disable --now zookeeper
systemctl disable --now cloud-agent
systemctl disable --now df-web-service
systemctl disable --now fauths
systemctl disable --now fpermit
systemctl disable --now fuser
systemctl disable --now talker
systemctl disable --now acl-controller
systemctl disable --now alarm
systemctl disable --now pcap
systemctl disable --now postman
systemctl disable --now report
systemctl disable --now analysis
systemctl disable --now diagnose
systemctl disable --now statistics
systemctl disable --now monitor
systemctl disable --now exchange
systemctl disable --now manager
systemctl disable --now warrant
systemctl disable --now trisolaris
systemctl disable --now genesis
systemctl disable --now droplet
systemctl disable --now pcap-rest
systemctl disable --now apm-server
systemctl disable --now influxdb
systemctl disable --now elasticsearch
systemctl disable --now kibana
systemctl disable --now telegraf
systemctl disable --now chronyd
systemctl disable --now influxdb-relay
systemctl disable --now rsyslog
systemctl disable --now reciter
systemctl disable --now mysqld
systemctl disable --now redis
systemctl disable --now openresty
systemctl disable --now salt-master
systemctl disable --now cleaner
systemctl disable --now cerebro
```

```
systemctl disable --now lake
systemctl disable --now httpd
systemctl disable --now webssh.service
systemctl disable --now clickhouse-server@0.service
systemctl disable --now hamgrd.service
```

v5.7.0

准备升级至 v5.7.0 之前的操作

- 升级 deepflow-release

挂载镜像

```
mount -o ro xxxx.iso /media/
```

执行升级

```
yum --disablerepo=* -y install /media/deepflow-release-*.rpm
```

v5.6.3

准备升级至 v5.6.3 之前的操作

- 挂载镜像/同步源

挂载镜像

```
mount -o ro xxxx.iso /media/
```

同步源

```
rsync -acqrz --delete [控制器主机名]:/media/{yum,pip,docker-image} /
usr/local/deepflow/
```

所有节点执行以下命令

```
yum clean all
```

- 升级 mntnct 和 saltstack 之前，需要先备份 pillar 文件夹

```
mv /usr/local/deepflow/saltstack/pillar /usr/local/deepflow/saltstack/pillar.bak
```

- mntnct 和 deepflow-saltstack 需提前升级

更新

```
yum --disablerepo=* --enablerepo=deepflow update -y deepflow-saltstack
```

```
yum --disablerepo=* --enablerepo=deepflow update -y mntnct
```

- 恢复 pillar 配置

1. 恢复 misc.sls 备份内的 salt-master 到 salt-mastes.sls
2. 恢复 redis/mysql/influxdb 账号密码等相关配置到对应的 sls
3. droplet/trisolaris/trident/dedicated-trident 等调优参数恢复

- 从控制器新增 cloud-agent 部署，需要提前在从控制器安装避免升级出错

安装

```
salt -N proxy_controller cmd.run "yum --disablerepo=* --enablerepo=deepflow install -y cloud-agent"
```

- 安装 web-tools、webssh、web-sched 服务

安装

```
yum --disablerepo=* --enablerepo=deepflow install -y web-tools
yum --disablerepo=* --enablerepo=deepflow install -y webssh
yum --disablerepo=* --enablerepo=deepflow install -y web-sched
```

- 安装 baseline 服务

主备控制器安装

```
yum --disablerepo=* --enablerepo=deepflow install -y baseline
# /etc/baseline.yaml 修改配置文件 statistics 与下面保持一致
statistics: http://127.0.0.1:20404
```

- 安装 log-cleaner 服务

主备从控制器安装

```
salt -N controller cmd.run "yum --disablerepo=* --enablerepo=deepflow install -y log-cleaner"
salt -N proxy_controller cmd.run "yum --disablerepo=* --enablerepo=deepflow install -y log-cleaner"
```

tsdb 和专属采集器

```
salt -N tsdb cmd.run "yum --disablerepo=* --enablerepo=deepflow install -y log-cleaner"
salt -N dedicated-trident cmd.run "yum --disablerepo=* --enablerepo=deepflow install -y log-cleaner"
```

- 使用 openresty 代替 httpd
 - 5.6.3 部署使用 openresty 作为 yum/pip 源服务器，由于 httpd 与 openresty 具有相似的功能，多引入组件将会增加运维成本（例如：修复漏洞），由于 httpd 出镜像源外无其他作用，因此移除 httpd

主备控制器停止

```
systemctl stop httpd
```

主备控制器增加配置文件/usr/local/openresty/nginx/conf/conf.d/source.conf

```
server {
    listen      23333;
    listen      [::]:23333;
    server_name _;

    location ~* ^/(yum|pip) {
        rewrite /$1/(.*) /$2 break;
        root /usr/local/deepflow/$1;
        autoindex on;
    }
}
```



```

        autoindex_exact_size off;
        autoindex_localtime on;
    }

    access_log /var/log/openresty/source.access.log main;
    error_log /var/log/openresty/source-error.log error;

}
# 主备控制器重启 openresty
systemctl restart openresty

```

v5.6.2

准备升级至 v5.6.2 之前的操作

- 挂载镜像/同步源

```

# 挂载镜像
mount -o ro xxxx.iso /media/
# 同步源
rsync -acqrz --delete [控制器主机名]:/media/{yum,pip,docker-image} /
usr/local/deepflow/
yum clean all

```

- mntnct 和 deepflow-saltstack 需提前升级

```

# 更新
yum --disablerepo=* --enablerepo=deepflow update -y deepflow-saltstack
yum --disablerepo=* --enablerepo=deepflow update -y mntnct

```

- 在如下路径寻找 trident.sls.rpmnew,并覆盖更新 trident.sls

```

# 路径
/usr/local/deepflow/saltstack/pillar/

```

- 在如下路径寻找 nodegroup.rpmnew 并覆盖更新 nodegroup

```

# 路径
/usr/local/deepflow/saltstack/

```

- 根据部署手册, 执行 config 命令, 进行设备信息添加, 需注意所有 master 角色需同时指定为 elasticsearch-warm

```
mt saltstack.config xxx
```

警告: 此处注意保留原来配置的其他角色

- 新增 daemon apm-server, 需要提前安装避免升级出错

安装

```
yum --disablerepo=* --enablerepo=deepflow install -y apm-server
```

- 在 5.6.2 版本中 hamgrd 和 zookeeper 为必要组件，如果原环境中没有安装，需要提前安装避免升级出错

安装

```
yum --disablerepo=* --enablerepo=deepflow install -y hamgrd
```

```
yum --disablerepo=* --enablerepo=deepflow install -y zookeeper
```

- 主从控制器新增 droplet 部署，需要提前在主从控制器安装避免升级出错

安装

```
salt -N controller cmd.run "yum --disablerepo=* --enablerepo=deepflow install -y droplet"
```

```
salt -N proxy_controller cmd.run "yum --disablerepo=* --enablerepo=deepflow install -y droplet"
```

- 数据节点需要卸载 roze、stream，防止升级后 droplet 启动失败

先备份/etc/roze.yaml 和/etc/stream.yaml

如果有手动修改过 droplet, roze, stream 的配置(非 salt 渲染的配置项)，升级后需要将这些配置手动修改到 droplet.yaml，例如：roze 的配置项 store-queue-size, disable-second-write 等

```
salt -N tsdb cmd.run "yum --disablerepo=* --enablerepo=deepflow remove -y roze"
```

```
salt -N tsdb cmd.run "yum --disablerepo=* --enablerepo=deepflow remove -y stream"
```

v5.6.1

准备升级至 v5.6.1 之前的操作

- 5.6.1 授权新增字段控制器数量，若证书没有此字段会导致控制器进入运维状态，需要在升级前申请新的证书(需要配置控制器数量)导入数据库

导入操作如下所示（密码以 xxx 为例）

```
/usr/bin/mysql -uroot -pxxx -e "insert into deepflow.license(status, name,value) values(0,'filename',x'\`hexdump -v -e '1/1 \"%02x\"' xxxxx xxxxxxxxxxxxxxxxxxx.LIC`')"
```

- 确认 patch——5.6.0/02-20200529-12483 中以下命令已执行

需执行语句

```
mysql -uroot -p -D deepflow -e "ALTER TABLE alarm_policy ADD COLUMN target_line_full TEXT;"
```

确认方法：以下语句执行结果中包含 target_line_full 字段

```
mysql -uroot -p -D deepflow -e "show create table alarm_policy"
```

- 由于数据库版本升级，deepflow rpm 依赖于 mysql，因此必须先在主备控制器分别执行以下命令卸载 mysql 后重新安装软件，并恢复 nodegroup，然后才能执行升级命令

挂载镜像/同步源

```
rsync -acqrz --delete [控制器主机名]:/media/{yum,pip,docker-image} /usr/local/deepflow/
yum clean all
```

修改pip源

```
sed -i '/prefix/d' /etc/pip.conf
```

备份nodegroup

```
mv /usr/local/deepflow/saltstack/nodegroup /usr/local/deepflow/saltstack/nodegroup.bak
```

备份pillar

```
mv /usr/local/deepflow/saltstack/pillar /usr/local/deepflow/saltstack/pillar.bak
```

卸载mysql

```
yum erase -y mysql-community-client mysql-community-common mysql-community-devel mysql-community-libs mysql-community-libs-compat mysql-community-server mysql57-community-release
```

重新安装

```
yum --disablerepo=* --enablerepo=deepflow install -y acl-controller alarm analysis cleaner cloud-agent deepflow-database deepflow-saltstack df-web-service diagnose exchange fauths fpermit fuser genesis manager metering mntnct postman talker trisolaris
```

恢复nodegroup

```
mv /usr/local/deepflow/saltstack/nodegroup.bak /usr/local/deepflow/saltstack/nodegroup
```

恢复pillar

```
rm -rf /usr/local/deepflow/saltstack/pillar
```

```
mv /usr/local/deepflow/saltstack/pillar.bak /usr/local/deepflow/saltstack/pillar
```

v5.6.0

准备升级至 v5.6.0 之前的操作

- 监控网段页面已不再支持，需要记录录入的监控网段，升级完成后录入为子网
- 5.6.0 之后版本 cerebro 使用单独的用户和组运行，在升级前要创建用户和组，在主备控制器上执行如下操作

```
groupadd -r cerebro
useradd -g cerebro -r --shell /bin/false -c "cerebro user-daemon" cerebro
```

- Elasticsearch 从 5.2.2 升级至 6.8.8，需要先暂停 ES 的操作，在主控制器上执行如下操作

```
curl -XPUT "localhost:20042/_cluster/settings" -d '{"persistent":{"cluster.routing.allocation.enable": "primaries"}}' -H "Content-Type: application/json"
```

执行以下命令到没有 *failed* 为止

```
curl -XPOST "localhost:20042/_flush/synced"
```

- 5.6.0 依赖 cython 版本升级

```
python3 -m pip install -U pip
```

- 5.6.0 如下进程废弃，需要在升级前手动删除：

```
salt -N controller cmd.run 'yum --disablerepo=* -y remove guard'
salt -N controller cmd.run 'yum --disablerepo=* -y remove resource'
salt -N controller cmd.run 'yum --disablerepo=* -y remove screen'
salt -N controller cmd.run 'yum --disablerepo=* -y remove dashboard'
salt -N baseline cmd.run 'yum --disablerepo=* -y remove baseline'
salt -N tsdb cmd.run 'systemctl stop influxdb@disk_only'
salt -N tsdb cmd.run 'systemctl disable influxdb@disk_only'
salt -N tsdb cmd.run 'yum --disablerepo=* remove -y zero influxdb@disk_only'
salt -N analyzer cmd.run 'systemctl stop pcap-rest'
salt -N analyzer cmd.run 'yum --setopt=tsflags=noscripts remove -y pcap-rest'
salt -N controller cmd.run 'yum --disablerepo=* remove -y collapsar'
```

- 5.6.0 修改 baseline 为空
 - 修改 `/usr/local/deepflow/saltstack/nodegroup` 配置文件，将 baseline 配置为空列表
- 5.6.0 版本 `cloud-agent.yaml` 的 vsphere 云平台新增 `ipv4_netmask`、`ipv6_netmask` 配置项

```
vcenter:
  ipv4_netmask: 24
  ipv6_netmask: 64
```

- 所有控制器、数据节点（原分析器）执行如下命令启用 IPv6

```
sysctl -w net.ipv6.conf.all.disable_ipv6=0
```

软件升级

v5.6.1 及以前版本

5.6.1 及以前版本的升级

环境检查

警告： HA 环境下，目前暂且可能会出现备控制器 hamgrd 先于主控制器 hamgrd 启动，且时间差距超出 10 秒的情形，这种情形下可能会导致升级过程中主备切换，因此建议通过控制口 ip 完成升级。

在升级前需要对系统环境进行检查，避免因环境导致升级 ERROR，检查环境如下：
- 查看所有节点的 salt-minion 进程状态，如果进程有 ERROR 输出，需要重启 salt-minion, 若 salt-minion 重启，salt-master 也需要重启 - systemctl status salt-minion
- systemctl status master - 测试 saltstack 是否工作正常 - salt-run manage.up # 检查活跃的 salt-minion 端，执行完成后，会输出所有 salt-minion 节点，如输出中缺少某个节点，证明此节点无法与 salt-master 通信 - salt * pillar.get
'master:nodegroups' # 获取 pillar 参数，测试 saltstack 工作是否正常，如无法获取数据，请执行下面命令 - salt * saltutil.sync_all # 同步 salt 所有配置信息

离线升级

将安装包拷贝到控制器上，执行如下命令更新控制器软件包：

```
mount -o ro deepflow-offline-release-XXX.iso /media  
##### 避免因升级触发 HA 切换 (5.5.4 版本及以后无需此操作，已包含在升级过程中)  
salt -N controller service.stop hamgrd  
/media/install.sh -u
```

更新完后，在如下路径寻找.rpmnew，如果存在，更新对应的配置文件并删除.rpmnew。注意：如果修改了 deepflow-master.conf 或 nodegroup，需要重启 salt-master。

```
/usr/local/deepflow/saltstack/  
/usr/local/deepflow/saltstack/pillar/  
/usr/local/deepflow/conf/deepflow.conf  
/etc/
```

继续执行如下命令完成升级：

```
mt saltstack.upgrade mount_root=/media  
umount /media
```

注意：完成后重启控制器上 systemctl list-dependencies deepflow.target 中的如下服务：

```
for daemon in "talker manager acl-controller statistics analysis pcap t  
risolaris alarm report exchange diagnose genesis"; do
```

```
salt -N controller cmd.run "systemctl restart $daemon"
done
```

在线升级 [仅用于测试环境相同版本升级]

使用 Jenkins 打包机或自动化部署测试机作为升级源，目前 HOSTNAME 可以选择 10.30.49.10。

```
mt saltstack.upgrade ssh=$HOSTNAME:/usr/local/deepflow/
```

完成后，如果需要修改或修改了 deepflow.conf，需要重启控制器上 systemctl list-dependencies deepflow.target 中的所有服务。

数据库升级 [仅用于测试环境相同版本升级]

连接 mysql 后，执行如下命令

```
use deepflow
source /usr/local/deepflow/db/issu/5.x.x.sql
```

完成后，需要重启控制器上 systemctl list-dependencies deepflow.target 中的如下服务：

```
for daemon in "talker manager acl-controller statistics analysis pcap t
risolaris alarm report exchange diagnose genesis"; do
    salt -N controller cmd.run "systemctl restart $daemon"
done
```

ES 索引 mapping 升级

- 停止所有数据节点上的 stream 进程
- 更新 es 索引 mapping，在控制器上执行如下命令（注意：当天的索引 mapping 不会被更新，除非删除索引）

```
mt elasticsearch.initialize
# 提示是否删除索引，选择不删除
```

- 重新添加 kibana 中的 dfi_flow__0_*索引
- 启动所有数据节点上的 stream 进程

升级问题处理及检查

- 如果在升级过程中遇到数据节点内核没升级，可以通过如下命令进行单独升级，升级完成后重启数据节点，使新内核生效

```
salt -S <NODE_IP> cmd.run 'uname -r' // 确定内核版本是否为 4.9.90-29.
el7.x86_64
salt -S <NODE_IP> state.apply kernel
```

- 若升级执行 `mt saltstack.upgrade` 时报如下错误，需要重启 `salt-master` 并刷新 `salt`，再重新执行升级操作

Rendering SLS 'base:roles' failed: Jinja variable 'dict object' has no attribute 'elasticsearch-warm'

```
systemctl restart salt-master
salt \* saltutil.sync_all
mt saltstack.upgrade
```

- 升级执行 `mt saltstack.upgrade` 时，遇到 `python36-PyYAML` 升级失败报错的解决方法

错误内容如下：

报错主机名：

```
Delta RPMs disabled because /usr/bin/applydeltarpm not installed.
Error unpacking rpm package python36-PyYAML-3.13-1.el7.x86_64
error: unpacking of archive failed on file /usr/lib64/python3.6/site-packages/PyYAML-3.13-py3.6.egg-info: cpio: rename failed - Is a
directory\npython36-PyYAML-3.12-1.el7.x86_64 was supposed to be re
moved but is not!'
```

执行如下命令后，重新执行 `mt saltstack.upgrade` 进行升级。

```
# salt -L [报错节点主机名列表，以逗号分隔] cmd.run "rm -rf /usr/lib64/
python3.6/site-packages/PyYAML-3*"
salt -L analyzer3,analyzer4 cmd.run "rm -rf /usr/lib64/python3.6/si
te-packages/PyYAML-3"
```

- 如果升级前曾做过页面定制，需要将定制过的内容备份，升级后重新配置。

v5.6.2

5.6.2 版本的升级

- 挂载镜像/同步源

```
# 挂载镜像
mount -o ro xxxx.iso /media/
# 同步源
rsync -acqrz --delete [控制器主机名]:/media/{yum,pip,docker-image} /
usr/local/deepflow/
yum clean all
```

- 执行升级后原有的 `redis` 密码会被覆盖，若需要保留旧密码，请修改以下文件中 `redis` 的 `password`：

```
/usr/local/deepflow/saltstack/pillar/misc.sls
```

- 在主控制器上执行如下命令，完成所有 `mt saltstack.list` 中节点的软件部署：

```
# 若执行时报错xxx 节点 is not connected:
# salt \* test.ping 检查该节点是否通信正常
# salt-key -L 检查密钥交互是否正常
# 若有异常可尝试重启 salt-minion 后再执行此命令
mt deepflow.upgrade

# 若执行时在某一步骤报错, 问题解决后可以根据报错结尾提示(例如 stage=upgrade_init, upgrade_base)
# 执行 upgrade 命令时带上该参数(例如 mt deepflow.upgrade stage=upgrade_init, upgrade_base)
# upgrade 即可从报错步骤处继续往下执行
mt deepflow.upgrade stage=xxxx,xxxx
```

- 在主备控制器执行如下命令重启 redis：

```
systemctl restart redis
```

v5.6.3 及之后版本

5.6.3 及之后版本的升级

- 挂载镜像/同步源

```
# 挂载镜像
mount -o ro xxxx.iso /media/
# 同步源
rsync -acqrz --delete [控制器主机名]:/media/{yum,pip,docker-image} /usr/local/deepflow/
yum clean all
```

- 执行升级后原有的 redis 密码会被覆盖，若需要保留旧密码，请修改以下文件中 redis 的 password：

```
/usr/local/deepflow/saltstack/pillar/redis.sls
```

- 在主控制器上执行如下命令，完成所有 `mt saltstack.list` 中节点的软件部署：

```
# 若执行时报错xxx 节点 is not connected:
# salt \* test.ping 检查该节点是否通信正常
# salt-key -L 检查密钥交互是否正常
# 若有异常可尝试重启 salt-minion 后再执行此命令
主区域升级命令
    mt deepflow.upgrade
从区域升级命令
    mt deepflow.upgrade region=slave
```



```
# 若执行时在某一步骤报错, 问题解决后可以根据报错结尾提示( 例如 stage=upgrad
e_init,upgrade_base)
# 执行upgrade 命令时带上该参数( 例如mt deepflow.upgrade stage=upgrade_
init,upgrade_base)
# upgrade 即可从报错步骤处继续往下执行
主区域升级命令
    mt deepflow.upgrade stage=xxxx,xxxx
从区域升级命令
    mt deepflow.upgrade region=slave stage=xxxx,xxxx
```

- 在主备控制器执行如下命令重启 redis:

```
systemctl restart redis
```

v5.7.3

5.7.3 版本升级

- 使用最新 ISO 容器化重新部署 DeepFlow ， 参阅 K8s 部署模式篇
- 下载并解压 01-patch 到指定目录
 - tar -zxvf xxx_patch.tar.gz -C /usr/local/deepflow/patch
- MySQL 数据恢复及升级 bash # 进入 mntnct 服务对应的 POD 执行


```
kubectl exec -it -n deepflow pod-xxxxxxx sh mysql -hmysql -
uroot -p -P30130 deepflow < /usr/local/deepflow/patch/deepflow.sql
mysql -hmysql -uroot -p -P30130 deepflow <
/usr/local/deepflow/patch/5.7/5.7.3/01-20211104-16397-
FEATURE/mysql_issu.sql
```
- 退出 Pod 环境, 进入宿主环境, 卸载 DeepFlow 组件再重新部署
 - 卸载: /usr/local/deepflow/bin/deepflow-deploy -ed
 - 再安装: /usr/local/deepflow/bin/deepflow-deploy -d
- 进入 mntnct 的 pod, 并进入 /usr/local/deepflow/patch 中具体 Patch 目录, 执
 行 post_patch 脚本
 - bash /usr/local/deepflow/patch/5.7/5.7.3/01-20211104-16397-
 FEATURE/post_patch.sh
- 待 Pod 全都 Running 后登入前端, 检查功能是否正常

v6.0.0 及之后版本

6.0.0 版本升级

- 升级前需先备份
 - mv /usr/local/deepflow/templates

/usr/local/deepflow/templates_\$(date +%Y%m%d)_bak
 - mv /usr/local/deepflow/bin /usr/local/deepflow/bin_\$(date

+%Y%m%d)_bak

- 跨版本升级必须使用 ISO 先重装一遍（参考《基础部署》），但无需 -j 初始化数据库
 - 注意：6.0.0 提供两个参数模板，values.yaml 为默认参数，不要修改。任何改动需按原格式写入到 values-custom.yaml 中，便于保存和备份环境自定义参数。
- 从 OSS 中获取 Patch 压缩包
- 执行新版本的 00 号 Patch
- /usr/local/deepflow/bin/deepflow-patch [Patch 包路径]

example:

```
/usr/local/deepflow/bin/deepflow-patch /root/15-20211213-16728-BUG.tar.gz
```

- 待 Pod 全都 Running 后登入前端，检查功能是否正常
- 继续执行升级后操作

升级后操作

v6.1.0

升级到 v6.1.0 之后的操作

主从区域互访配置更新

- 如存在主从区域配置，参考 values.yaml 中 customResource 部分，重新配置主从区域访问相关参数

v6.0.1

升级到 v6.0.1 之后的操作

授权升级

- 注意：v5 授权在升级到 v6.0.1 后将不可用，需要重新申请 v6 的授权文件
- v6 使用全新授权机制，分为流量分发、网络监控、应用监控三大功能，需分别进行授权。
- 升级 v6 授权的相关操作如下：
 - 根据环境信息申请新的授权文件，需要重点关注环境中所需授权的具体功能
 - 在系统-授权管理-授权信息页面，上传新的授权文件并删除已有授权文件
 - 在系统-授权管理-采集器授权详情页面，按需修改采集器的授权功能

重新导入 Grafana Dashboards

执行如下命令

先卸载

```
helm uninstall -n deepflow grafana-dashboards
```

再重装

```
/usr/local/deepflow/bin/deepflow-deploy -uo grafana-dashboards
```

专属采集器、ESXi 采集器配置更新

- 新增专属采集器、ESXi 采集器自定义配置参数块
- 如环境存在专属采集器、ESXi 采集器配置，需参考 values.yaml 中默认参数，重新配置 values-custom.yaml 中相关配置，再使用 -uo dedicated-trident/-uo trident 将配置应用进集群。

v6.0.0

升级到 v6.0.0 之后的操作

视图升级

- 视图导出功能需要进行视图升级，升级后方可使用视图导出功能，6.0.0 版本新建视图不受此影响。
- 修改 df-web-core 的 configmap `kubectl -n deepflow edit configmap/df-web-core`
- 修改 configmap 的 VIEW_UPDATE 为 true
- 重启 `kubectl rollout restart deploy/df-web-core-deployment -n deepflow`
- 到视图页面，点击右上角视图升级按钮，即可把当前视图内所有子视图升级
- 重复所有需要升级的视图页面

升级客户采集器

v5.7.2

升级到 v5.7.2 之后的操作

设置 ClickHouse 系统日志保留时长为 30 天

- 设置数据节点 ClickHouse 实例 0 的系统日志保留时长，执行：

```
salt -N tsdb cmd.run "clickhouse-client -q 'ALTER TABLE system.query_log MODIFY TTL event_date + INTERVAL 30 DAY '"
salt -N tsdb cmd.run "clickhouse-client -q 'ALTER TABLE system.trace_log MODIFY TTL event_date + INTERVAL 30 DAY'"
salt -N tsdb cmd.run "clickhouse-client -q 'ALTER TABLE system.query_thread_log MODIFY TTL event_date + INTERVAL 30 DAY'"
salt -N tsdb cmd.run "clickhouse-client -q 'ALTER TABLE system.metric_log MODIFY TTL event_date + INTERVAL 30 DAY'"
salt -N tsdb cmd.run "clickhouse-client -q 'ALTER TABLE system.asynchronous_metric_log MODIFY TTL event_date + INTERVAL 30 DAY'"
```

- 设置数据节点 ClickHouse 实例 1 的系统日志保留时长（如果不存在实例 1，则无须执行），执行：

```
salt -N tsdb cmd.run "clickhouse-client --port 9001 -q 'ALTER TABLE
system.query_log MODIFY TTL event_date + INTERVAL 30 DAY'"
salt -N tsdb cmd.run "clickhouse-client --port 9001 -q 'ALTER TABLE
system.trace_log MODIFY TTL event_date + INTERVAL 30 DAY'"
salt -N tsdb cmd.run "clickhouse-client --port 9001 -q 'ALTER TABLE
system.query_thread_log MODIFY TTL event_date + INTERVAL 30 DAY'"
salt -N tsdb cmd.run "clickhouse-client --port 9001 -q 'ALTER TABLE
system.metric_log MODIFY TTL event_date + INTERVAL 30 DAY'"
salt -N tsdb cmd.run "clickhouse-client --port 9001 -q 'ALTER TABLE
system.asynchronous_metric_log MODIFY TTL event_date + INTERVAL 30
DAY'"
```

Influxdb 停止监听 20049 端口

- 停止监听 20049 端口，执行：

```
salt -N controller state.sls influxdb.config
salt -N controller state.sls influxdb.stop
salt -N controller state.sls influxdb.start
```

视图升级

- 对视图中告警策略进行升级
- 需要修改主控制器上 /var/www/lcweb/public/app_config_user.js 的 VIEW_UPDATE 为 true
- 到视图页面，点击右上视图升级按钮，即可把当前视图内所有的告警策略升级
- 重复所有需要升级的视图页面

v5.7.1

升级到 v5.7.1 之后的操作

配置 ClickHouse

- 若 5.7.0 时数据存储双副本，5.7.1 也要存储双副本
 - 需要确保 zookeeper 状态正常，否则不支持双副本
 - 查看 zookeeper 状态，执行 echo ruok|nc 127.0.0.1 2181 看是否返回 imok(若不支持 nc 可执行 yum install nc -y --disablerepo=* --enablerepo=deepflow 安装)
 - 需要保证该区域 master 节点个数至少 3 个，数据节点(tsdb)个数至少 2 个
 - mt saltstack.list 查看 roles 中分别包含 master 和 tsdb 的节点个数
 - 配置 mt saltstack.config clickhouse_0_tiers=\$path1
clickhouse_1_tiers=\$path2, 必需配置 2 个路径

- 并且配置/usr/local/deepflow/saltstack/pillar/droplet.sls 文件的 metrics_replica_enabled 为 true
- 若 5.7.1 存储只须单副本
 - 配置 mt saltstack.config clickhouse_0_tiers=\$path1, 只需配置 1 个路径即可
- 配置完成后执行如下命令:


```
salt -N tsdb state.apply tsdb.stop
salt -N tsdb state.apply clickhouse.install
salt -N tsdb state.apply clickhouse.config
salt -N tsdb state.apply clickhouse.start
salt -N tsdb state.apply tsdb.config
salt -N tsdb state.apply tsdb.start
salt -N tsdb state.apply telegraf.config
salt -N tsdb state.apply telegraf.stop
salt -N tsdb state.apply telegraf.start
```

全景图数据源配置更新

- 进入系统->数据节点->全景图存储配置页面
- 对于全景图原始数据源(1s,1m)若修改过默认保留时长, 升级后需要重新设置下保留时长
- 对于自定义数据源, 需要删除再重建, 名字需保持一致, 否则影响数据迁移
 - 注意: 5.7.1 版本只支持基于 1s,1m 的原始数据源创建自定义数据源。
 - 若存在自定义数据源基于数据源 1h 创建, 需要修改为基于 1m 的数据源创建。
- 对于流量日志的保留时长, 如果 570 修改过默认值(3 天), 也需要重新设置下

采集器组配置更新

- 检查容器-V/容器-P 类型采集器配置的同步资源信息是否为开启; 如果是开启, 则需要置为关闭。

流量日志默认限速改变

- 5.7.0 流量日志默认写入限速是 1k 条/秒, 5.7.1 增大为 50k 条/秒
- 若流量日志写入达到最大值, 每天的写入数据将占用超过 500G 磁盘空间
- 请评估流量日志的写入增多对磁盘容量和磁盘 IO 的影响, 并作出相应调整, 如:
 - 扩容磁盘
 - 调整流量日志的保留时间: 登录管理平台, 进入系统->数据节点->全景图存储配置->数据源列表->流量日志->操作进行修改
 - 调整流量日志写入限速: 参考功能介绍->全景图->流量搜索->功能规格->4.5. 流日志写入限速

5.7.0 到 5.7.1 数据迁移

预处理和说明

- 确保升级后基本业务正常
- 确保全景图数据源配置更新步骤已完成
- 不支持全景图秒级数据的迁移
- 不支持流量日志数据的迁移

数据迁移操作

1. 迁移工具说明

- 升级后，默认数据节点已部署迁移工具
- 可执行文件路径: `/usr/bin/influxdb-migrate`
- 配置文件路径: `/etc/influxdb_migrate.yaml`
 - 迁移速率设置: `limit-write-rate` 默认限制 100000 条每秒, 可根据实际情况调整.
- 迁移日志: `/var/log/influxdb_migrate.log`

2. 登录所有数据节点

3. 分钟级数据迁移配置

- 默认配置如下，将自动迁移所有分钟级数据。

```
migrates-1m:
- dbs:                                # 迁移的数据库列表. 流统计: 包含 vtap_flow, vtap_flow_edge, vtap_flow_port, vtap_flow_edge_port 子数据库
  - vtap_flow
  - vtap_flow_edge
  - vtap_flow_port
  - vtap_flow_edge_port
  time-range: ""                      # 迁移数据的时间范围, 如只迁移最近 1 天的数据: "time > now()-1d and time < now()", 默认: "", 表示迁移所有数据
- dbs:                                # 包统计: 包含 vtap_packet, vtap_packet_edge 子数据库
  - vtap_packet
  - vtap_packet_edge
  time-range: ""
- dbs:                                # 广域网: 包含 vtap_wan, vtap_wan_port 子数据库
  - vtap_wan
  - vtap_wan_port
  time-range: ""
- dbs:
```

```
- vtap_acl
time-range: ""
```

4. 自定义数据源数据迁移配置

- 默认为空, 不迁移自定义数据源数据, 需要根据实际情况具体配置。
- 如果小时/天的数据基于分钟的数据聚合而成, 则在分钟数据迁移的时间范围内, 小时/天的数据无须再迁移。
 - 例如: 分钟迁移的时长是最近 7 天, 则小时/天数据的迁移的时长 `time-range` 应设置为 "`time < now()-7d`"
- 如果天的数据基于小时的数据聚合而成, 则在小时数据迁移的时间范围内, 天的数据无须再迁移。
 - 例如: 小时迁移的时长是最近 30 天, 则天迁移的时长 `time-range` 应设置为 "`time < now()-30d`"

配置示例:

migrates-1h1d:

`dbs` 配置说明:

- 流统计: 包含 `vtap_flow`, `vtap_flow_edge`, `vtap_flow_port`, `vtap_flow_edge_port` 子数据库

- 包统计: 包含 `vtap_packet`, `vtap_packet_edge` 子数据库

- 广域网: 包含 `vtap_wan`, `vtap_wan_port` 子数据库

- **dbs:** # 例如: 迁移`流统计`名称为`1h`的按`小时`聚合的自定义数据源

```
- vtap_flow
- vtap_flow_port
- vtap_flow_edge
- vtap_flow_edge_port
```

rp: 1h # 数据源名称: web 界面显示的名称(实际底层存储会使用 `rp_1h`)

ck-table: 1h # 一般和 `rp` 相同, 除非要迁移后改名

unsummable: avg # 非累加聚合的值, 只支持: `avg`, `max`, `min`

interval: 1h # 聚合的时间粒度, 只支持: `1h`, `1d`

time-range: "`time < now()-7d`" # 迁移 7 天前的数据, # 迁移数据的时间范围, 如只迁移最近 1 天的数据: "`time > now()-1d and time < now()`", 默认: "", 表示迁移所有数据

- **dbs:** # 例如: 迁移`流统计`名称为`1d`的按`天`聚合的自定义数据源

```
- vtap_flow
- vtap_flow_port
- vtap_flow_edge
- vtap_flow_edge_port
```

rp: 1d

ck-table: 1d

unsummable: avg

```
interval: 1d
time-range: "time < now()-30d" # 迁移30 天前的数据
```

5. 迁移执行

```
influxdb-migrate
# 若迁移数据量大, 可使用后台执行
# nohup influxdb-migrate > influxdb-mig.Log 2>&1 &
```

6. 迁移后操作

- 验证迁移后, 可以正确查询到历史数据信息
- 若数据节点不同时为控制器, 停止 influxdb 服务, 清理 influxdb 数据

```
systemctl stop influxdb
yum remove influxdb -y --disablerepo=* --enablerepo=deepflow
# 找到influxdb 的存储目录
cat /etc/influxdb/influxdb.conf |grep "TSM storage" -A2
# 删除目录数据
```

v5.7.0

升级到 v5.7.0 之后的操作

- 5.7.0 提升了 MySQL 的安全等级, 使用 MySQL8 默认密码策略, 使用更复杂的 MySQL 密码
 - 升级后 pillar 内的 mysql 密码将被替换, 使用 mt deepflow.reconfig 刷新各个进程配置文件的密码
 - 将 root 用户的密码修改为 pillar 内的密码, 参考“部署扩容-组件配置文件-开源组件密码配置-MySQL 修改密码”章节
- 5.7.0 进程日志的记录直接使用 es client 写入 es。在 /etc/web/ 路径寻找 .rpmnew 文件, 如果存在, 复制 elasticsearch 节点数据更新到对应的配置文件中, 并配置正确的 host-list、port、user_name、user_password 后重启以下服务。

```
systemctl restart fauths
systemctl restart fuser
systemctl restart df-web-service
systemctl restart web-tools
```

警告: 如果进程的配置文件 .yaml 中没有 elasticsearch 节点信息, 进程会使用如下默认配置来和 es 服务通信。如果系统调整过 es 的配置, 建议按照上文更新 yaml 文件, 并重启服务。

```
host-list:['127.0.0.1']
port:20042
user_name:None
user_password:None
```


v5.6.3

升级到 v5.6.3 之后的操作

- 完成告警适配 topN 报警：

- 停止 alarm 服务
- 删除旧版本索引模板

```
curl -X DELETE 'http://127.0.0.1:20042/_template/dfi_alarm_event'
```

请求结果：

```
{"acknowledged":true}
```

- 删除旧版本索引

```
curl -X DELETE 'http://127.0.0.1:20042/dfi_alarm_event*'
```

请求结果：

```
{"acknowledged":true}
```

- 启动 alarm 服务

- 日志索引变更处理

- 主控制器上执行如下操作初始化流日志，HTTP 日志和 DNS 日志索引：

```
# 执行下面命令，若返回"Remove all incorrect indices? [y/N]"，请输入'y'  
mt elasticsearch.initialize
```

- 确认 droplet 写流日志的限速值为 1000

- 在控制器将 saltstack 的 pillar 中的 throttle 配置项的值设置为 1000。

```
# 查看当前 throttle 值，若是 1000，则无须处理  
cat /usr/local/deepflow/saltstack/pillar/deepflow.sls | grep throttle  
# 若值非 1000，则设置为 1000，并刷新 tsdb 节点的配置  
sed -i 's/ throttle:./ throttle: 1000/g' /usr/local/deepflow/saltstack/pillar/deepflow.sls  
salt-run state.orch tsdb_run
```

- 参考各类型采集器中的[采集配置](#)章节矫正采集网口、隧道解封装的配置，以展现虚拟网络链路追踪能力

- 注意：K8s 容器环境下，不再需要采集 cni0/flannel.1/tunl0 接口的流量，需要将此类接口从配置中去除

- 升级到 v5.6.3R3 及以后版本，需要手动调整各采集器组配置，避免默认配置被高亮显示

- 具体调整操作：如果配置项是使用默认配置，则将对配置项清空

- 各配置默认值:

类别	配置	单位	空值(默认值)
资源限制	内存限制	M 字节	768
	CPU 限制	逻辑核	1
	分发流限速	Mbps	1000
	采集包限速	Kpps	200
	分发熔断监控间隔	秒	10
	分发熔断阈值	Mbps	0
	日志发送速率	条/秒	300
	日志打印等级		INFO
基础配置参数	采集网口		^tap.*
	采集包长	字节	65535
	流量采集方式		自适应
	解封装隧道类型		无
	虚拟机 MAC 解析		接口 MAC
	虚拟机 XML 文件		/etc/libvirt/qemu/
	最长同步间隔	秒	60
	最长逃逸时间	秒	3600
	裸 UDP 最大 MTU	字节	1500
	裸 UDP 外层 VLAN		0
	是否请求 NAT IP		否
	日志存储时长	天	30
全景图配置参数	数据套接字		UDP
	PCAP 套接字		UDP
	HTTP 日志代理客户端		X-Forwarded-For
	HTTP 日志 TraceID		X-B3-TraceId
	应用层日志解析包长	字节	256

	流日志采集速率	10000
	应用层日志采集速率	10000
包分发配置参数	分发套接字	裸 UDP
	内层附加头	无
基础功能开关	同步资源信息	关闭
	日志发送	开启
全景图功能开关	指标数据	开启
	非活跃端口指标数据	开启
	应用层指标数据	开启
	包统计数据类型	播送类型
	流统计秒级数据	开启
	包统计秒级数据	开启
	过滤流日志	全部
	过滤应用层日志	全部
包分发功能开关	全局去重	开启

- 若将之前已存在的从控制器，改为其他从区域的主节点控制器，需要将此控制器的区域主节点标识改为是，具体操作如下所示：

```
use deepflow;
update controller set node_type=1 where ip="控制器 ip";
```

- 使用超管账户登录系统后通过 系统->配置管理->web 配置 来添加文件传输模块默认配置项 `file_storage`、`radius_account_switch`。参考文档 部署扩容->控制器与数据节点部署->页面定制配置 中的 web 安全配置 章节，来添加缺失的 `file_storage`、`radius_account_switch` 配置节点。如果不添加该参数，系统会默认使用硬编码的配置项（和该节点的配置参数一致）。
- Windows 系统 trident 升级
 - 新的功能要求 trident 安装路径不包含空格，如之前安装在默认路径（`C:\Program Files\trident`）或带了空格的路径中，需要先卸载再安装。
- 升级到 v5.6.3R3 及以后版本, 各区域性能统计数据本地化存储处理以及支持监控进程启停

- 开放 MySQL 的 grafana 用户访问权限，使从区域的 grafana-server 可以访问
- 从区域 InfluxDB 创建 telegraf 数据库和启动 grafana-server
- 新增部署 procsnoopd 服务和重新渲染 monitor 和 droplet 的配置并重启相关服务

主区域主控制器处理

```
salt -N controller cmd.run "mysql -e \"update mysql.user set host='%' where user='grafana'; flush privileges;\""
salt \* state.sls telegraf.install
salt \* state.sls telegraf.start
salt-run state.orch monitor_config
salt \* cmd.run "systemctl restart telegraf; systemctl restart influxdb-relay; systemctl start procsnoopd"
salt-run state.orch tsdb_config
salt -N tsdb cmd.run "systemctl restart droplet"
```

从区域主节点处理

```
tsdb-dump -e "create database telegraf"
tsdb-dump -e "alter retention policy autogen on telegraf duration 168h shard duration 24h default"
salt-run state.orch monitor_install
salt-run state.orch monitor_config
salt-run state.orch monitor_start
salt \* cmd.run "systemctl restart telegraf; systemctl restart influxdb-relay; systemctl start procsnoopd"
salt-run state.orch tsdb_config
salt -N tsdb cmd.run "systemctl restart droplet"
systemctl start grafana-server
```

- SSL/TLS 协议信息泄露漏洞(CVE-2016-2183)修复

主区域主控制器处理

```
salt -N controller state.sls openresty
```

从区域主节点处理

```
salt -N controller state.sls openresty
```

v5.6.2

升级到 v5.6.2 之后的操作

- 在控制器上远程将 KVM/Workload 类型的采集器从 v5.6.1 升级到 v5.6.2
 - v5.6.2 后为了减少采集器访问控制器需要开放的端口号，默认不再提供 HTTP 的升级方式，需要手动修改配置才能完成将 v5.6.1 的 trident 升级到 v5.6.2；升级方法如下：

- 设置新版本 trident 可执行文件路径，执行 `ln -sf /usr/sbin/trident /usr/local/deepflow/yum/trident`，注意必须是软链接
- 执行 `trident -v` 获取版本号

```
trident -v
# 352-d3b8c80b756f1261410d0fc847db6d7955324ce2 2019-05-22
```

其中,revision 为 352-d3b8c80b756f1261410d0fc847db6d7955324ce2

- 修改主控制器上
/usr/local/deepflow/saltstack/pillar/trisolaris.sls 中的如下配置:

```
# x.x.x.x 为主控制器 IP
self_update_url: http://x.x.x.x:23333/yum/trident
trident_revision: 352-d3b8c80b756f1261410d0fc847db6d7955324ce2
```

- 确认采集器和 self_update_url 配置的 IP 地址网络通的
 - 执行 `salt -N controller saltutil.refresh_pillar`
 - 执行 `salt -N controller state.sls deepflow_config.trisolaris`
 - 执行 `salt -N proxy_controller state.sls deepflow_config.trisolaris`
- Windows 环境 Trident 升级还需执行如下步骤（纯 linux 环境可以忽略该步骤）：
 - 注意：原版本和升级目标版本所对应的 DeepFlow 版本都不能低于 5.5.8
 - 将安装包 trident-setup.exe 在任意一台 windows 上解压
 - 打开命令行窗口进入 trident.exe 所在目录，执行 `.\trident.exe -v` 查看版本号，确保 revision 和 linux 的保持一致
 - 将 trident.exe 上传到控制器的 /usr/sbin/ 目录下
 - `chmod 777 /usr/sbin/trident.exe` 增加权限
 - 执行 `cp /usr/sbin/trident.exe /usr/local/deepflow/yum/`
- 执行 `mt vtap.upgrade name=XXX` 升级对应的 vtap
- 采集器原先写到 trisolaris.yaml 文件内的配置转移到页面，需要在采集器配置页面重新进行配置，否则采集器可能无法正常工作
- v5.6.1 到 v5.6.2 的 tsdb 数据迁移
 - 登录所有数据节点执行以下操作

```
# 停止当前数据库，备份数据，并启动备份数据库，然后再启动当前数据库
tsdb_issu_562.sh
```

```
# 请等待备份数据库启动成功，若如下命令成功返回，则表示备份数据库启动成功
tsdb-dump -p 30044
```

```
# 将数据从备份数据库，迁移到当前数据库
influx-migrate-562
# 迁移日志: /var/log/influxdb/influx_migrate.log
# 注意: 默认限制每秒迁移20000条记录，可通过`/etc/influxdb/influxdb_migrate.yaml`的`limit-write-rate`调整。
```

- 登录所有单独的控制节点(不同时是数据节点)执行以下操作

```
# 停止当前数据库，备份数据，并启动备份数据库，然后再启动当前数据库
tsdb_issu_562.sh
# 无须进行数据迁移，直接停止备份数据库
tsdb_issu_562.sh stop
```

- 迁移结束, 验证业务正常后, 清理原数据

```
# 停止备份数据库
tsdb_issu_562.sh stop
# 删除备份数据
tsdb_issu_562.sh clean
```

- 流日志索引变更处理

- v5.6.2 版本需要执行如下操作初始化流日志索引, 否则流日志无法写入:

```
mt elasticsearch.initialize
```

- 对于 v5.6.1 写入的流日志的清除方法:

- 登录 `http://<deepflow>/cerebro/`, 选择 DeepFlow Cluster
- 搜索 `dfi_flow__` 的所有索引文件, 删除即可。

v5.6.1

升级到 v5.6.1 之后的操作

- v5.6.0 到 v5.6.1 的 tsdb 数据迁移

- 登录所有数据节点执行以下操作

```
# 停止当前数据库，备份数据，并启动备份数据库，然后再启动当前数据库
tsdb_issu_561.sh
```

请等待备份数据库启动成功, 若如下命令成功返回, 则表示备份数据库启动成功

```
tsdb-dump -p 30044
```

将数据从备份数据库, 迁移到当前数据库

```
influx-migrate-561
```

迁移日志: /var/log/influxdb/influx_migrate.log

注意: 默认限制每秒迁移10000条记录, 可通过`/etc/influxdb/influxdb\_migrate.yaml`的`limit-write-rate`调整.

- 迁移结束, 验证业务正常后, 清理原数据

停止备份数据库

```
tsdb_issu_561.sh stop
```

删除备份数据

```
tsdb_issu_561.sh clean
```

- v5.6.1 已不再支持共享服务器类型的采集器（数据节点对应的采集器），需要在系统-采集器页面手动删除这部分采集器，若这部分采集器在接收交换机镜像流量或 NetFlow/sFlow，需要再手动录入为专属服务器类型的采集器。

- cloud-agent.yaml 添加新配置项

- 华为私有云新加 zones 配置项，用于限制对接的资源范围

```
huawei_private:
```

```
  zones:
```

```
    # - az_name1
```

```
    # - az_name2
```

- Kubernetes 新加 token 配置项，支持用 token 对接资源；新加 port_name_regex 配置项，用于学习节点上符合正则的端口

```
kubernetes:
```

```
  token: ''
```

```
  port_name_regex: '^(?=(cni|flannel|cali|tunnel))'
```

- 5.6.1 开始全景图数据保留时长改为在页面配置，请确认页面显示的时长与所有数据节点中 InfluxDB 的时长一致

- 在数据节点上执行 tsdb-dump 可查看所有数据库，全景图数据包含 flow_flow、vtap_packet、vtap_wan 开头的数据库

- 依次对数据库查询保留时长的信息,脚本如下:

```
db=(vtap_flow vtap_flow_edge vtap_flow_edge_port vtap_flow_p
ort vtap_packet vtap_packet_edge vtap_wan vtap_wan_port)
for db in ${db[*]} ; do
  echo $db
```

```
tsdb-dump -d $db -e rp
```

```
done
```

```
# 结果示例:
```

```
# Influxdb rp_1m: http://127.0.0.1:20044
```

```
# name      duration shardGroupDuration replicaN default
```

```
# ----      -
```

```
# autogen 0s      168h0m0s      1      false
```

```
# rp_1m    168h0m0s 17h0m0s      1      true
```

```
# rp_1s    36h0m0s  3h0m0s      1      false
```

- rp_1m 所在行 duration 列表示分钟数据保存时长，rp_1s 所在行表示秒数据保存时长
- 若不一致，需要从页面上修改存储时间保留时长直至 InfluxDB 显示正确
- 若页面初始显示为 7 天且希望保留 7 天，需要先修改为 6 然后再改为 7
- rp_1m 所在行 shardGroupDuration 列的值是否处于 17h~24h, 若不在该范围，请在数据节点执行如下脚本改正。

```
db=(vtap_flow vtap_flow_edge vtap_flow_edge_port vtap_flow_port vtap_packet vtap_packet_edge vtap_wan vtap_wan_port)
```

```
for db in ${db[*]} ; do
```

```
    echo $db
```

```
    tsdb-dump -e "alter retention policy rp_1m on $db shard duration 17h"
```

```
tsdb-dump -d $db -e rp
```

```
done
```

- 删除/etc/roze.yaml 中的 influxdb-default-retention-policy 配置项的所有内容。

v5.6.0

升级到 v5.6.0 之后的操作

- 将 v5.5.8 的监控网段录入为子网，注意设置为正确的区域、可用区、VPC、类型属性
- 完成 ES 升级：
 - 使用 systemctl 重启所有节点的 ES
 - 登录 Cerebro，等待旧索引变成 yellow 或 green
 - 在主控制器上执行以下操作


```
curl -XPUT "localhost:20042/_cluster/settings" -d '{"persistent":{"cluster.routing.allocation.enable": null}}' -H "Content-Type: application/json"
```

- 检查 kibana 是否能正常使用，否则先暂停 kibana，在 Cerebro 删除.kibana 索引，最后重启 kibana

- v5.5.8 到 v5.6.0 的 tsdb 数据迁移

- 登录所有数据节点执行以下操作

```
influx-migrate
# 迁移日志: /var/log/influxdb/influx_migrate.log
# 注意: 默认限制每秒迁移10000条记录, 可通过`/etc/influxdb/influxdb_migrate.yaml`的`limit-write-rate`调整.
```

- 迁移结束, 验证业务正常后, 清理原数据

```
influx-migrate -clean
```

- 完成告警适配新版本 ES:

- 停止 alarm 服务
- 删除旧版本索引模板

```
curl -X DELETE 'http://127.0.0.1:20042/_template/dfi_alarm_event'
```

请求结果:

```
{"acknowledged":true}
```

- 删除旧版本索引

```
curl -X DELETE 'http://127.0.0.1:20042/dfi_alarm_event*'
```

请求结果:

```
{"acknowledged":true}
```

- 启动 alarm 服务

- cloud-agent.yaml 添加新配置项

- 华为私有云新加 zones 配置项，用于限制对接的资源范围

```
huawei_private:
  zones:
    # - az_name1
    # - az_name2
```

- Kubernetes 新加 token 配置项，支持用 token 对接资源；新加 port_name_regex 配置项，用于学习节点上符合正则的端口

```
kubernetes:
  token: ''
  port_name_regex: '^(?=(cni|flannel|cali|tunnel))'
```

Patch 更新

概览

容器化部署模式下，DeepFlow 通过 K8s 管理所有状态。因此 Hot Patch 一般以更新容器镜像方式提供。pre_patch 及 post_patch 需进入 mntnct 容器内执行。

Patch 升级方法

制作 Patch 的注意事项：- pre_patch.sh：可选，若存在必须可执行，内含升级 Patch 之前需要执行的内容。- 容器镜像：可选，必须是小版本 tag - post_patch.sh：可选，若存在必须可执行，内含升级 Patch 之后需要执行的内容。

打 Patch 的注意事项：- 不支持在 x86/arm 控制器上远程升级 arm/x86 环境的采集器。- 请保持 patch 目录的路径与 gitlab 上一致，例如路径后缀需要形如 /5.6/5.6.3-R3/01-20201022-13434-BUG，否则会影响 DeepFlow 版本信息生成。- 确保 values-custom.yaml 中的 DeepFlow Version 信息符合要求 e.g. “DeepFlow-V6.0.1” “DeepFlow-V6.0.1-01” “DeepFlow-V6.0.1-21”

自动升级流程

仅支持 6.0.0 及以后版本

功能

- 支持容器镜像及 post_patch.sh、mysql_issu.sql 的升级
- 支持专属采集器、ESXI 采集器自动升级
- 不支持 trident kvm 采集器自动升级
- 聚合 Patch 自动升级（需使用聚合 Patch 包）

升级流程

- 获取 Patch 压缩包 / 聚合 Patch 压缩包
- 执行 patch 命令

```
Usage: bash deepflow-patch [--allpatch] [Patch File Path]
                        [Patch File Path]          # 直接使用 Patch 文
                        件路径作为第一个参数 e.g. /root/5.7.3_09-20211126-16616-BUG.tar.gz
                        -k, --quick                  # 多个 Patch 时可使
用 -k 参数关闭 helm 升级操作，仅最后一个 Patch 时再统一升级，可减少 Pat
ch 时间
                        -a, --allpatch              # 聚合 Patch 包升级
模式，必须使用聚合 Patch 包
                        -h, --help                  # help how to use
```

```
# 单个普通 Patch 升级
/usr/local/deepflow/bin/deepflow-patch 15-20211213-16728-BUG.tar.gz
# 多个普通 Patch 快速升级(-k 参数关闭本次 helm 升级操作, 仅最后一个 Patch 时再统一升级, 可减少 Patch 时间)
/usr/local/deepflow/bin/deepflow-patch --quick 15-20211213-16728-BUG.tar.gz
# 聚合 Patch 一键升级
/usr/local/deepflow/bin/deepflow-patch --allpatch 15-20211213-16728-BUG.tar.gz
```

注意: - mntnct pod 需运行在拥有 kubectl 的 master1 上, 多控制器节点场景如遇 mt 在其他控制器 node 上, 可使用 kubectl cordon/uncordon node 禁止调度/开放调度命令临时禁止调度到其他节点, 重启 pod 后实现迁移 mt 运行节点。- 从区域升级 Patch 暂需手动升级, 不执行 post_patch.sh 及 mysql issu

手动升级流程

- 下载并解压 01-patch 到指定目录
 - tar -zxvf xxx_patch.tar.gz -C /usr/local/deepflow/patch
- 合并 patch 中的 registry 文件夹与原 registry, 无需 mv 原 registry, 参考如下命令, 注意修改成正确路径
 - /bin/cp -rvf /usr/local/deepflow/patch/5.7/5.7.3/02-20211118-16589-BUG/registry/* /usr/local/deepflow/registry/
- 如有部署模板更新, 需按照路径合并, 举例:
 - /bin/cp -rvf /usr/local/deepflow/patch/5.7/5.7.3/02-20211118-16589-BUG/templates/* /usr/local/deepflow/templates/
- 根据 patch_image_tag_list.txt 中的镜像 tag 修改 /usr/local/deepflow/templates/values.yaml 组件对应 tag
- MySQL 数据升级 bash # 进入 mntnct 服务对应的 POD 执行 kubectl exec -it -n deepflow pod-xxxxxxx sh mysql -hmysql -uroot -p P30130 deepflow < /usr/local/deepflow/patch/5.7/5.7.3/02-20211118-16589-BUG/mysql_issu.sql
- 进入 mt 容器环境, 并进入 /usr/local/deepflow/patch 中具体 Patch 目录, 执行 post_patch 脚本
 - bash /usr/local/deepflow/patch/5.7/5.7.3/01-20211104-16397-FEATURE/post_patch.sh
- 修改 values.yaml 中的 deepflowVersion, 尾部增加 “-{最新 patch 号}”, 示例: “deepflowVersion: DeepFlow-V5.7.3-14”
- 安装 DeepFlow 相关组件 /usr/local/deepflow/bin/deepflow-deploy -d
- 升级 DeepFlow 相关组件 /usr/local/deepflow/bin/deepflow-deploy -u
- 重启 df-web kubectl rollout restart -n deepflow deploy/df-web-deployment
- 待 Pod 全都 Running 后登入前端, 检查功能是否正常, 检查版本号是否正确

服务器上下线

新增备控制器

参考高可用初始化。

删除备控制器

暂不支持。

新增数据节点（也作为从控制器）

参考基础部署，分为如下步骤：

- 服务器升级至 Python 3
- 执行 `saltstack.add` 和 `saltstack.config` 录入数据节点部署信息
- 执行 `mt deepflow.deploy` 执行录入服务器的部署
- 重启服务器使得新的内核生效

删除数据节点（也作为从控制器）

如删除 IP 为 172.20.1.171 节点

- 执行 `mt saltstack.remove ip=172.20.1.171` 删除该节点
- 执行 `mt deepflow.refresh_config` 刷新配置

修改控制器 ip 和 hostname（仅支持 allinone）

如修改 IP 为 192.168.14.2，修改 hostname 为 df-master，控制器 root 密码为 xxx

- 执行 `mt saltstack.modify-controller ip=192.168.14.2 hostname=df-master passwd=xxx` 修改配置信息
- 执行 `mt deepflow.reconfig` 刷新配置

数据备份

控制器 MySQL 数据备份

通过 DeepFlow 页面系统-控制器列表可导出控制器上包含 MySQL、配置文件等数据。也可使用下述方法手动到备份 MySQL。

控制器数据主要在 MySQL 中，参考 MySQL 标准备份方法即可，备份之前请先查看 MySQL 数据目录大小、并停止控制器上的如下 MySQL 写入进程：- talker - exchange - manager - trisolaris

恢复数据之前也需要停止上述 MySQL 写入进程。

备份命令

```
mysqldump --databases deepflow grafana --set-gtid-purged=OFF > /tmp/deepflow.sql
```

数据节点指标数据备份

指标数据存储于 InfluxDB 中，可参考如下方法进行备份和恢复。

```
influxd backup
  [ -database <db_name> ]
  [ -portable ]
  [ -host <host:port> ]
  [ -retention <rp_name> ] | [ -shard <shard_ID> -retention <rp_name> ]
]
  [ -start <timestamp> [ -end <timestamp> ] | -since <timestamp> ]
  <path-to-backup>
```

```
influxd restore [ -db <db_name> ]
  -portable | -online
  [ -host <host:port> ]
  [ -newdb <newdb_name> ]
  [ -rp <rp_name> ]
  [ -newrp <newrp_name> ]
  [ -shard <shard_ID> ]
  <path-to-backup-files>
```

数据节点流量日志数据备份

流量日志数据使用 Elasticsearch 存储，可使用快照方式进行备份：

```
PUT /_snapshot/my_backup
{
  "type": "fs",
  "settings": {
    "location": "my_backup_location"
  }
}
```

数据节点 PCAP 数据备份

PCAP 存储在文件中，直接以备份文件的方式进行数据备份，注意排除后缀为 .tmp 的文件，这样的文件正在进行写入。

整体卸载

警告： 此操作仅用于测试环境。

DeepFlow 无法直接从新版本降级至旧版本，也不能直接从 master 版本切换到待发布版本。但是可以先卸除掉所有从 Deepflow 安装的组件，然后再完成一次全新部署。

需要注意的是，在卸载 DeepFlow 组件时，相关配置文件会被重命名为.rpmsave。
可以运行如下命令：

```
#### 当 salt 可用时，在所有节点执行命令可以在主控制器上使用 salt '*' cmd.run  
'CMD' 来简化操作  
#### 当 salt 可用时，在特定角色节点执行命令可以在主控制器上使用 salt -N NODEG  
ROUP cmd.run 'CMD' 来简化操作
```

```
#### 在所有节点卸载 Yunshan RPM  
\rpm -qa --queryformat "%{NAME} %{VENDOR} %{PACKAGER}\n" | awk "/Yunsha  
n/ {print \$1}" | xargs yum --disablerepo=* -y remove  
rm -rf /usr/local/deepflow
```

```
#### 在所有节点卸载 python3  
yum --disablerepo=* -y remove python3*  
rm -rf /usr/local/lib64/python3.6/site-packages  
rm -rf /usr/lib64/python3.6/site-packages  
rm -rf /usr/local/lib/python3.6/site-packages  
rm -rf /usr/lib/python3.6/site-packages
```

```
#### 在主区域 controller 节点卸载 mysql  
yum remove mysql  
rm -rf /var/lib/mysql  
rm -rf /var/lib/mysql  
rm -rf /usr/lib64/mysql
```

```
#### 在 tsdb 节点卸载 tsdb  
yum --disablerepo=* -y remove influxdb
```

```
#### 在 elasticsearch-warm 节点卸载 es  
yum --disablerepo=* -y remove elasticsearch rsyslog-elasticsearch
```

```
#### 在所有区域 controller 节点卸载 monitor  
yum --disablerepo=* -y remove grafana telegraf
```

```
#### 在所有节点卸载 monitor  
yum --disablerepo=* -y remove influxdb-relay
```

```
#### 在所有节点卸载 salt  
yum --disablerepo=* -y remove salt-*  
rm -rf /etc/salt
```

超时配置

修改 statistics API 超时时间

- 修改主备控制器上的文件
/usr/local/deepflow/statistics/venv/lib/python3.6/site-packages/sanic/config.py

```
self.REQUEST_TIMEOUT = 60 // API 请求超时时间, 单位 s  
self.RESPONSE_TIMEOUT = 60 // API 响应超时时间, 单位 s
```

- 修改主备控制器上的文件/etc/statistics.yaml

```
clickhouse:  
    timeout: 60 // 查询 clickhouse 超时时间, 单位 s
```

- 执行 `systemctl restart statistics`, 重启进程

修改页面请求超时时间

- 修改主备控制器上的文件/var/www/lcweb/public/app_config_user.js

```
API_TIMEOUT: 60 * 1000, // 请求 API 超时时间, 单位 ms
```

- 执行 `ctrl+F5` 强制刷新页面