微服务平台 TSF

常见问题

腾讯云

版权所有：腾讯云计算（北京）有限责任公司

【版权声明】

©2013-2025 腾讯云版权所有

本文档（含所有文字、数据、图片等内容）完整的著作权归腾讯云计算（北京）有限责任公司单独所有，未经腾讯云事先明确书面许可，任何主体不得以任何形式复制、修改、使用、抄袭、传播本文档全部或部分内容。前述行为构成对腾讯云著作权的侵犯，腾讯云将依法采取措施追究法律责任。

【商标声明】

腾讯云

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。未经腾讯云及有关权利人书面许可，任何主体不得以任何方式对前述商标进行使用、复制、修改、传播、抄录等行为，否则将构成对腾讯云及有关权利人商标权的侵犯，腾讯云将依法采取措施追究法律责任。

【服务声明】

本文档意在向您介绍腾讯云全部或部分产品、服务的当时的相关概况，部分产品、服务的内容可能不时有所调整。
您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或默示的承诺或保证。

【联系我们】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100或95716。

文档目录

常见问题
- 功能和概念相关
- 购买和计费相关
- 资源管理相关
- 应用管理相关
- Spring Cloud 应用接入相关
- Mesh 应用相关
- 日志服务相关
- JVM 监控相关
- 镜像相关
- 子账号使用相关
- 其他问题

版权所有：腾讯云计算（北京）有限责任公司

常见问题
功能和概念相关

最近更新时间：2021-11-10 12:58:17

TSF 有哪些功能?

TSF 提供应用全生命周期管理、数据化运营、立体化监控和服务治理等功能。TSF 拥抱 Spring Cloud 、Service Mesh 微服务框架，帮助企业客户解决传统集中式架构转型的困难，打造大规模高可用的分布式系统架构，实现业务、产品的快速落地。更多详情请参考 产品功能 。

TSF 有哪些版本?

TSF 有基础版、专业版和铂金版，不同版本支持的功能和服务会有差异，对应的价格也会不同。

- 基础版：包括基本的应用生命周期管理、应用监控、调用链与日志等功能。
- 专业版：包含了基础版的所有功能，同时支持服务限流、路由、熔断等服务治理功能。
- 铂金版：包含了基础版和专业版的所有功能，同时支持微服务网关、全链路灰度等高级能力。

详细说明参考 产品系列 和 计费概述 文档。

TSF 与 TKE 的关系是什么?

TKE（Tencent Kubernetes Engine）是基于原生 Kubernetes 提供以容器为核心的、高度可扩展的高性能容器管理服务。TSF 是以微服务为核心的服务治理平台，用户可以使用云服务器或者容器来部署微服务，其中容器集群管理和容器应用部署使用了 TKE 提供的服务。

应用和服务之间是什么关系?

在 TSF 中应用是管理一组程序包、镜像、配置的抽象概念，服务表示已经注册到注册中心的程序。可以从以下维度来区分应用和服务：

概念	程序类型	主要功能	与部署组的关系
应用	无特定类型	程序包版本管理、配置版本管理	部署组使用应用中的程序包和配置
服务	注册到注册中心的微服务	服务注册、服务限流、服务路由等服务治理功能	在相同命名空间下具有相同服务名的部署组属于一个服务

TSF 支持 Sentinel 流控相关的组件应用吗?

暂不支持。TSF 自身提供了替代能力，具体请参考 服务熔断原理及使用 和 服务限流原理及使用 。

购买和计费相关

最近更新时间：2021-11-10 12:57:03

TSF 如何收费?

- TSF 收费依据：根据不同版本和最大节点数收费。
- TSF 计费模式：预付费按月收费，直接购买12个月可享受83折优惠。
- TSF 购买方式：线上购买，详情请参考 购买方式 。

关于不同版本的功能和计费说明请参考 计费概述 和 产品系列 。

TSF 如何扩大规格?

TSF 支持的最大节点数为2000，如需申请更高规格请 提交工单 。

购买了 TSF 之后，是否需要另行购买 CVM?

TSF 是分布式应用的构建和管理平台，本身不包含 CVM 资源，您需要另行购买 CVM。购买之后，TSF 能够自动同步您账号下的 CVM 资源并进行使用和管理。

资源管理相关

最近更新时间：2022-06-20 14:34:45

云服务器加入容器集群后，项目属性为何变为"默认项目"?

由于加入到 TSF 容器集群中的云服务器 CVM 实例可能属于不同项目，TSF 将加入到容器集群中的云服务器 CVM 的项目属性统一修改为**默认项目**。如果需要集群内云服务器分布在不同的项目，请自行前往云服务器控制台迁移项目。

虚拟机集群内云主机的可用状态为何显示不可用?

虚拟机集群中的云服务器显示"不可用"状态是因为 agent 无法连接 TSF 后台服务器导致的，因此需要检查 agent 的可用状态。

1. 首先检查云服务器的状态是否为"运行中"。如果云服务器的状态不是运行中，请确保云服务器为开机运行状态。
2. 登录云服务器，执行 `ps aux | grep 'agent'` 命令查看是否有包含 `tsf-agent` 命名的进程。如果没有，请查看步骤3。
3. 切换到 `/root` 目录下，查看是否有 `tsf_agent` 目录。
   - 如果没有，需要将主机从集群中删除后，再重新导入。
   - 如果有，进入 `/root/tsf_agent/ops` 目录下，先执行 `uninstall.sh` ，再执行 `install.sh` 命令。然后通过 ps 命令检查进程。

如何手动恢复 tsf-agent 进程?

tsf-agent 有 crontab 做进程检查和恢复，如果 crontab 被停用、卸载、修改会导致 tsf-agent 宕机无法自恢复。手动恢复 tsf-agent 的方法如下：

1. 检查 tsf-agent 的安装目录是否存在，通常在 /root/tsf-agent。
2. 检查 tsf-agent 的安装用户， ls -l tsf-agent/，通常是 root，那以 root 身份到安装目录下执行 ./ops/start.sh。

TSF 容器集群数达到上限如何处理?

默认情况下每个账号在一个地域最多能创建5个容器集群，如需扩大额度，请 提交工单 给 **容器服务 TKE** 产品。

TSF 不同集群内的服务是否支持相互调用?

在网络连通性的前提下，同一命名空间内的服务可以相互发现和相互调用。如果要实现跨集群的服务访问，有两个前提：

1. 集群内实例网络互通
2. 集群关联相同的命名空间。多个集群通过命名空间名称（而不是命名空间 ID）作为关联的 key。您可以在命名空间一级页面中看到关联的集群信息。

举例：有2个集群分别是 cls-a 和 cls-b，两个集群内实例网络互通（如在相同 VPC 内），并且都关联了命名空间 test-ns。要实现跨集群访问，需要确保在创建部署组 provider（cls-a集群内）和 consumer（cls-b集群内）时，使用相同的命名空间 test-ns。

TSF 如何设置开发环境和测试环境?

在同一个账号下，您可以通过集群或者命名空间来划分开发环境和测试环境：

- **使用集群划分**：创建两个集群并使用不同的 VPC 作为集群网络，通过网络实现服务之间隔离的作用。使用集群来划分的特点是不同集群不会共享底层的云服务器等资源。
- **使用命名空间划分**：创建两个非全局命名空间，不同命名空间内的服务默认不支持相互调用。

集群网络和容器网络如何设置?

集群网络与容器网络是集群的基本属性。通过设置集群网络和容器网络可以规划集群的网络划分。

- **集群网络**：为集群内主机分配在节点网络地址范围内的 IP 地址，您可以选择私有网络中的子网用于集群的节点网络，更多私有网络介绍请参考 私有网络和子网 。

- **容器网络：** 为集群内容器分配在容器网络地址范围内的 IP 地址，您可以自定义三大私有网段作为容器网络， 根据您选择的集群内服务数量的上限，自动分配适当大小的 CIDR 段用于 kubernetes service，同时容器网络自动为集群内每台云主机分配一个24位的网段用于该主机分配 Pod 的 IP 地址。

详细说明请参考 集群及容器网络设置 。

CVM 集群应用的日志清理机制是什么？

- 若不设置日志转储，会在30天后自动删除。
- 若选择日志转储至 CKafka（参考 日志投递 ），则可以存储自定义的时长。具体时长由用户购买的 CKafka 实例规格决定。

应用管理相关

最近更新时间：2024-01-19 11:48:11

TSF 是否支持在同一台服务器上安装多个应用?

在 TSF 中，应用的部署有两种类型：

- CVM 云服务器独占实例：在一台 CVM 云服务器上，仅部署单独一个应用。通常根据应用需要的资源配置来购买 CVM 云服务器。
- 容器实例：TSF 使用 Docker 容器在一台独立的 CVM 云服务器上创建多个 Docker 实例，允许在每一个 Docker 实例上部署一个应用。

如何查看实例的 Agent 版本?

1. 在**部署组**页面，单击目标部署组“ID/部署组名”，进入服务实例详情列表。
2. 在目标实例操作栏单击**查看agent版本**即可查看实例的 Agent 版本。

如何将实例 Agent 升级至最新版本?

- 虚拟机部署场景：您需要把实例移出集群后，再重新移入集群，重新部署服务实例。
- 容器部署场景：您需重新编写 dockerfile，生成新的镜像后重新部署服务实例，详情请参见 制作容器镜像 。

TSF 应用实例状态为什么显示 Agent 异常?

TSF Agent 会定期上报心跳数据给 TSF 管理模块，如果 Agent 停止上报状态，则某段时间后该机器将会被判定为未知状态。通常而言，该问题是由于 Agent 停止导致。
您可以尝试在云服务器界面，重启该云服务器。

重启服务器后 TSF Agent 是否能自动重启?

是的，重启服务器后 TSF Agent、应用都会自动重启。

容器部署组执行部署操作时提示内存（或 CPU）不足?

请检查该部署组所在集群和命名空间中的节点的内存（或 CPU）的使用情况，确保在执行部署操作时填写的内存（或 CPU）数值小于剩余内存（或 CPU）资源 。
您可以在集群的节点列表页面中找到已分配 CPU 和已分配内存信息。

如何排查应用是否部署失败?

1. 在应用详情页，单击部署组操作列的**查看日志**查看 stdout 日志，通过日志初步定位是否是业务程序本身问题。如果没有日志信息，进行步骤2。

2. 单击**变更记录**，查看本次部署任务的 taskid。

3. 登录虚拟机或容器，查看本次任务的日志信息 `/root/tsf-agent/agent/task/<taskid>`，其中 taskid 是步骤2中的任务 ID。
4. 您可以通过日志信息初步定位部署失败原因，如果无法排查，可 提交工单 反馈任务日志信息。

程序包无法上传如何解决?

当发现程序包无法上传时，请检查浏览器是否设置了代理。您可以尝试换一个浏览器或者切换网络重新进行上传。

创建应用时，提示命名空间数达到配额如何解决?

您需要在容器服务控制台中，选择 **镜像仓库** > **我的镜像**，删除超过限额的命名空间。

删除应用后是否可以恢复?

不可以，删除应用操作不可逆，所有的数据都会被清除。

程序包容量达到上限时如何解决?

默认每个租户程序包的存储容量为100GB，如果超过上限，上传程序包时会提示错误信息。此时您需要删除仓库中历史版本的程序包后才能上传新的程序包。

如何选择容器部署组的访问方式?

请参见 容器应用部署组 。

将一台已在 CVM 创建的实例添加到 TSF 的时候，是否需要重装 Agent?

需要。由于系统是克隆的（虽然克隆前已安装 Agent），但实例 ID、Local IP 等信息还是之前系统的，导致 Agent 上报实例状态异常，所以克隆后的系统仍需要再次重装 Agent。

Spring Cloud 应用接入相关

最近更新时间：2024-01-22 10:55:51

TSF 支持哪些 Spring Cloud 版本?

TSF 目前支持 Spring Cloud Edgware、Spring Cloud Finchley、Spring Cloud Greenwich、Spring Cloud Hoxton 四个版本。

- Spring Cloud Edgware 对应 Spring Boot 1.5.x
- Spring Cloud Finchley 对应 Spring Boot 2.0.x
- Spring Cloud Greenwich 对应 Spring Boot 2.1.x
- Spring Cloud Hoxton 对应 Spring Boot 2.3.x

TSF 在开源 Spring Cloud 的基础上做了加强，用户只需要替换掉部分依赖并开启注解即可将改造后的应用部署在 TSF 平台上。更多关于 Spring Cloud 的兼容性说明请参考 Spring Cloud 概述 、SDK 下载 、SDK 版本更新日志 文档。

在工程的配置文件中，是否需要填写服务注册中心地址?

- 对于本地开发调试的应用，在启动 Java 应用的启动参数中需要填写轻量级服务注册中心 consul 的 IP 和 Port。配置文件（如 application.yml）中则无需填写。
- 对于通过 TSF 平台部署的应用，既不需要在启动参数中设置注册中心的地址，也不需要在配置文件中设置注册中心地址。SDK 会通过环境变量获取注册中心地址。

Mesh 应用相关

最近更新时间：2022-06-20 14:32:29

应用部署成功后，服务列表中为何没有出现服务？

请检查部署压缩包中是否包含了 `spec.yaml` ，且 `spec.yaml` 的格式是否正确。
如果不存在 `spec.yaml` 或者格式不对，TSF 无法将服务注册到服务注册中心。详情参考 TSF Mesh 开发指引 。

服务实例显示离线状态如何解决？

在 Mesh 环境下，TSF Sidecar 会定期通过调用服务的健康检查接口获取服务健康状态，并将健康状态上报到服务注册中心。由于某些原因，例如用户健康检查接口信息配置错误、端口配置错误、或者服务实例出现访问失败，则会导致服务不健康。
您可以通过以下步骤进行排查：

1. 查看服务配置信息

- 服务配置信息错误
  查看应用的软件包，获取服务配置信息（spec.yaml），检查服务名是否为期望暴露的服务名、端口号是否为服务真实监听的端口号、健康检查接口是否存在、检查健康接口格式是否正确（不含 ip:port，类似 `/health` 是符合的）。
- 服务配置文件格式错误
  将 spec.yaml 内容，复制到 yamllint 中，校验 yaml 格式是否正确。如果格式正确，则继续检查字段名称，是否与下面示例的格式一致。

```
apiVersion: v1
kind: Application
metadata:
name: service1
namespace: nsTester
spec:
services:
  - name: user
    healthCheck:
      path: /health
    ports:
      - targetPort: 8089
        protocol: http
```

- 服务配置没有挂载到正确的位置
- 在容器环境下，排查业务是否在容器的启动脚本中，把 spec.yaml 和 apis 目录（可选），复制到挂载路径 `/opt/tsf/app_config` 下面。
- 在虚拟机环境下，排查业务程序包根目录下面，是否存在 spec.yaml 以及 apis 目录（可选），如果没有，则需要修改。
- 通用排查方法：

  1.1 检查 pilot-agent 加载配置文件的绝对目录。
  输入： `curl 127.0.0.1:15020/config/agent|grep appConfigDir` ， 输出： `appConfigDir: /data/demo/shopService` 代表 pilot-agent 会从 `/data/demo/shopService` 中加载 spec.yaml。

  1.2 确认 appConfigDir 指向的目录存在 spec.yaml。

2. 查看健康检查返回

登录应用所在的容器或者虚拟机，通过调用本地的健康检查路径，查看返回码是否为200，如果不是200，证明服务存在健康问题。详情参考 TSF Mesh 开发指引 。

```
curl -i -H 'Host: local-service' {ip}:{Port}/<healthCheck_path>
```

3. 通过运维接口查看 sidecar 相关信息

登录应用所在的容器或者虚拟机，调用 pilot-agent 的运维接口，查看 sidecar 容器的相关状态信息。

- 查看接口列表
  调用 `GET 127.0.0.1:15020/help` 接口查看接口列表：

```
[root@TENCENT64 ~/pilot-agent/op]# curl 127.0.0.1:15020/help
admin commands are:
 GET /health: print out the health info for data-plane
 GET /config_dump/{component}: print out the configuration of the component, component
can be pilot-agent/envoy/mesh-dns
 GET /help: print out list of admin commands
 GET /config/agent: print out the pilot-agent configuration
 GET /config/services: print out the services info
 GET /config/global: print out the global mesh configuration
 GET /config/envoy: print out the envoy startup configuration
 GET /version: print out the pilot-agent version
 GET /version/{component}: print out the version of the component, component can be
pilot-agent/envoy/mesh-dns
 GET /latest_error: print out the latest error info
 GET /status: print out the status, result could be <INIT>, <CONFIG_READY>,
<FLOW_TAKEOVER>, <SERVICE_REGISTERED>
 GET /epoch/{component}: print out the component restart epoch, component can be
envoy/mesh-dns
 POST /hot_restart/{component}: hot restart the component process, component can be
envoy/mesh-dns
 PUT /update: Update the component
 PUT /logging/{scope}/{level}: dynamic change logging level, scope can be
healthz/admin/ads/default/model, level can be info/warn/error/none/debug
```

- 查看版本号
  调用 `GET 127.0.0.1:15020/version` 接口查看版本号：

```
[root@TENCENT64 ~/pilot-agent/op]# curl 127.0.0.1:15020/version
1.0.13.release-20190225_103608
```

- 查看 sidecar 健康状态
  调用 `GET 127.0.0.1:15020/health` 接口查看 sidecar 健康状态：

```
[root@TENCENT64 ~/pilot-agent/op]# curl 127.0.0.1:15020/health
{"envoy":{"status":"UP"},"mesh-dns":{"status":"UP"},"status":"UP"}
```

如果出现部件不健康的组件（status 不为 UP），或者接口调用失败，则需要通过 提交工单 联系后台运维人员处理。

4. 调用 clusters 接口

登录应用所在的容器或者虚拟机，调用 envoy 的 clusters 接口，查看本地 cluster（格式为 in#port#serviceName）是否存在或者健康状态是否 healthy。
输入：

```
curl http://127.0.0.1:15000/clusters
```

输出：

```
in#8080#reporttimeb::default_priority::max_connections::1024
in#8080#reporttimeb::default_priority::max_pending_requests::1024
in#8080#reporttimeb::default_priority::max_requests::1024
in#8080#reporttimeb::default_priority::max_retries::3
in#8080#reporttimeb::high_priority::max_connections::1024
in#8080#reporttimeb::high_priority::max_pending_requests::1024
in#8080#reporttimeb::high_priority::max_requests::1024
in#8080#reporttimeb::high_priority::max_retries::3
in#8080#reporttimeb::added_via_api::true
in#8080#reporttimeb::9.77.7.132:8080::cx_active::0
in#8080#reporttimeb::9.77.7.132:8080::cx_connect_fail::0
in#8080#reporttimeb::9.77.7.132:8080::cx_total::4
in#8080#reporttimeb::9.77.7.132:8080::rq_active::0
in#8080#reporttimeb::9.77.7.132:8080::rq_error::0
in#8080#reporttimeb::9.77.7.132:8080::rq_success::4
in#8080#reporttimeb::9.77.7.132:8080::rq_timeout::0
in#8080#reporttimeb::9.77.7.132:8080::rq_total::4
in#8080#reporttimeb::9.77.7.132:8080::health_flags::healthy
in#8080#reporttimeb::9.77.7.132:8080::weight::1
in#8080#reporttimeb::9.77.7.132:8080::region::
in#8080#reporttimeb::9.77.7.132:8080::zone::
in#8080#reporttimeb::9.77.7.132:8080::sub_zone::
in#8080#reporttimeb::9.77.7.132:8080::canary::false
in#8080#reporttimeb::9.77.7.132:8080::success_rate::-1
```

- 如果 cluster 不存在，则需要通过 提交工单 联系后台运维人员处理。
- 如果状态不为 healthy，则执行后续步骤。

5. 调用 config_dump 接口

登录应用所在的容器或者虚拟机，调用 envoy 的 clusters 接口，查看本地的路由配置信息：

```
curl http://127.0.0.1:15000/config_dump -o config.json
```

通过 vi 打开 config.json 文件，并查找 in#8080#reporttimeb（本地 cluster，格式为 in#port#serviceName），查看配置中的服务地址（address）、端口（port_value）是否正确。
健康检查信息 health_checks、熔断配置 circuit_breakers 是否正确。如果不正确，且确认服务没有被熔断，则需要通过 提交工单 联系后台运维人员处理。

```
"dynamic_active_clusters": [
```

```
    {
     "version_info": "2018-10-17T11:31:50+08:00",
     "cluster": {
      "name": "BlackHoleCluster",
      "connect_timeout": "1s"
     },
     "last_updated": "2018-10-16T19:31:41.792Z"
    },
    {
     "version_info": "2018-10-17T11:31:50+08:00",
     "cluster": {
      "name": "in#8080#reporttimeb",
      "connect_timeout": "5s",
      "hosts": [
       {
        "socket_address": {
         "address": "9.77.7.132",
         "port_value": 8080
        }
       }
      ],
      "health_checks": [
       {
        "timeout": "5s",
        "interval": "10s",
        "unhealthy_threshold": 2,
        "healthy_threshold": 2,
        "http_health_check": {
         "path": "/health"
        }
       }
      ]
```


如何联系后台运维人员?

您可以通过 提交工单 的方式联系后台运维人员。

您需要提供以下信息，方便运维人员快速定位问题。

- 问题：B 服务注册失败，已初步定位，原因在于下发的配置与服务配置不相符。
- 服务节点信息：A 调用 B 服务，B 服务192.168.3.1，A 服务192.168.2.1。
- 服务类型信息：A 服务为 springcloud，B 服务为 Mesh。

日志服务相关

最近更新时间：2024-01-22 10:55:51

为什么无法显示 stdout 日志?

对于使用容器部署的应用，在下述两种情况下 TSF 会采集 stdout 日志：

- **业务容器启动服务进程时不重定向 stdout。**
  此时可以登录上容器所在的机器，通过 `kubectl logs` 或者 `docker logs` 观察程序自身有无输出 stdout。
- **业务容器启动服务进程时将 stdout 重定向到 `/data/tsf_std/stdout/logs/sys_log.log` 。**
  此时可以登录上容器所在的机器，通过 `kubectl exec` 或者 `docker exec` 进入到容器中，观察上述 `sys_log.log` 文件有无内容。

为什么无法显示自定义的文件日志?

1. 检查日志配置项中的日志采集规则
   - 确认日志配置项中的**日志路径**和应用程序在配置文件中的日志路径相同。
   - 确认日志配置项中的**日志解析格式**和应用程序在配置文件中的日志解析格式相同。例如，如果日志配置项为默认日志配置项（Spring Boot 日志格式），而应用程序实际使用 logback 和自定义的日志 pattern，那么 TSF 无法采集日志。用户需要创建一个 logback 格式的日志配置项，然后关联到部署组。
2. 在日志配置项的详情页中，查看日志配置项是否已经发布到对应的部署组中。
3. 日志配置项修改后，虚拟机部署的应用会自动使用修改后的日志配置项来采集日志，容器部署的应用需要重启部署组（先停止部署组，再启动部署组）后才会使用修改后的日志配置项来采集日志。
4. 如果容器镜像的时区不对，则无法检索到日志，需要在 Dockerfile 中调整时区。

```
# GMT+8 for CentOS
RUN /bin/cp /usr/share/zoneinfo/Asia/Shanghai /etc/localtime
RUN echo "Asia/Shanghai" > /etc/timezone
```

为什么实时日志一直加载不出来?

1. 请确认日志配置项已正确配置。
2. 实时日志会有15s时延，即显示从15秒前开始打印的日志。如果一直在加载中，可以确认下最近15s有没有日志打印，如果没有则会一直到有新的日志打印才会显示。

能否关掉 tsf-task-schedule-xxx 线程或者日志?

这个名称的线程都是 TSF SDK 的 schedule 线程，关闭后会导致部分数据同步功能不正常（如服务治理、配置等）。
如果不需要看到日志，可以将对 package 或 class 的日志级别调高即可。

JVM 监控相关

最近更新时间：2023-10-18 14:36:03

JVM 监控功能为何无法使用?

JVM 监控依赖实例上安装的探针。探针版本较低时 JVM 能力不可用。

JVM 如何升级 tsf-agent?

升级操作：

- VM 场景：您需要把实例移出集群后，再重新移入集群，重新部署服务实例。
- 容器场景：您需重新编写 dockerfile，生成新的镜像后重新部署服务实例，详情请参见 制作容器镜像 。

另外， JVM 监控功能依赖 javaagent，jdk1.5 以后才引入了 javaagent 技术，所以您需使用 jdk1.6 及以上版本，建议您使用 jdk1.8。

> **注意**
>
> - TencentCloudJvmMonitor-x.x.x监听11339端口，请注意避免冲突。
> - 请在新建容器部署组时需要额外加上端口映射。

火焰图采集失败如何处理?

火焰图适用于在 CPU 利用率持续较高的情况下进行热点函数分析。
在如下场景下，火焰图采集可能失败：

- 数据量过大：目前火焰图的数据量上限为2MB，超出时会采集失败；此时，请缩短所选择的采集时间后，重新进行采集。
- 当前无热点函数：进程处于低 CPU 消耗状态，即当前无热点函数时，火焰图采集可能失败（无法采集到热点函数）；此时，可尝试延长采集时间后重新采集，或待服务请求量较大时再重新采集。
- 无法和进程建立连接：发生无法和实例连接等异常情况时，采集任务会执行超时；此时，请您检查实例的连接状态，确认状态正常后再重新采集。

为何无法查看 JVM 日志?

JVM 日志的采集依赖于 TSF 的 GC 日志配置。如果您自定义了 GC 日志配置，将会覆盖 TSF 的 GC 日志配置，这将导致无法在界面中查看 JVM 日志。

版权所有：腾讯云计算（北京）有限责任公司

镜像相关

最近更新时间：2024-11-05 16:13:52

协作者为何无法推送镜像到主账号的镜像仓库？

主账号需要将协作者与 `QcloudCCRFullAccess` 策略关联，具体操作请参考 获取访问授权 。

默认容器是 UTC 时区，和宿主机的时区不一致如何解决？

保证容器的时区和宿主机一致，避免调用链日志时间收集等有偏差。如果基础镜像时区是没调整过的，那么在编写 `Dockerfile` 时再调整时区，例如基础镜像是 centos，那么微服务 `Dockerfile` 增加以下配置，并重新 build 即可：

```
#GMT+8
RUN /bin/cp /usr/share/zoneinfo/Asia/Shanghai /etc/localtime
RUN echo "Asia/Shanghai" > /etc/timezone
```

协作者使用镜像仓库时，为何显示未开通仓库？

如果主账号未开通过镜像仓库，会提示如下图所示信息，此时需要主账号登录 TSF 控制台，开通镜像仓库。主账号开通镜像仓库后，协作者/子账号才能继续使用镜像仓库。

子账号使用相关

最近更新时间：2024-01-19 11:48:12

主账号发生无权限访问如何解决?

主账号在使用 TSF 功能时提示无权限错误，此时需要主账号授权 TSF 创建服务角色 TSF_QCSRole，详情参考文档 准备工作 。

子账号身份使用 TSF 发生无权限报错如何解决?

子账号在使用 TSF 前，需要主账号将子账号关联 `tsf_PassRole` 策略，具体操作请参考 子账号获取访问授权-授予 tsf_PassRole 策略 。

使用控制台时，提示“当前账号尚未获得操作权限[action]，请联系主账号授权”如何解决?

您可根据提示的 action 判断：

- 是否是 TSF 产品内的权限。若出现明显的 CVM、TKE、VPC 字样，则需要主账号在 CAM 策略 中，配置相关读写权限。
- 若根据 action 语义判断，属于 TSF 内的功能权限。您可参考 角色管理 ，申请主账号为您配置相应权限点。

子账号使用 TSF 时，弹窗提示 “您没有权限执行此操作”如何解决?

问题描述：
子账号使用 TSF 平台时，可能报错 “您没有权限执行此操作” ，显示如下：

问题分析：
在使用 TSF 平台时，需要调用其他云产品的接口获取一些信息，包含 VPC、CVM、API 网关、镜像仓库、腾讯云可观测平台、TKE 等云产品。
解决方法：
建议对子账号授权 tsf_PassRole 策略，如果没有，需要主账号或者具有 QcloudCamRoleFullAccess 策略的用户创建角色。
子账号使用 TSF 时，需要主账号授予 tsf_PassRole 策略。具体操作请参考 子账号获取访问授权-授予 tsf_PassRole 策略 。
要将角色（及其许可策略）传递至 TSF 服务，用户必须具有**传递角色**至服务的许可。这有助于管理员确保仅被批准的用户可配置具有能够授予许可的角色的服务。具体参考 子账号获取访问授权-授予访问其他云产品权限 。
最终子账号需被授予如下策略：

策略	是否必选	说明
tsf_PassRole	必选	手动创建。

QcloudCamSubaccountsAuthorizeRoleFull	可选	访问管理（CAM）子账号授权服务角色相关权限，包含子账号在授权服务角色过程中涉及的全部权限。
QcloudTSFFullAccess	可选	腾讯微服务平台（TSF）全读写访问权限。
QcloudCCRFullAccess	可选	与镜像仓库相关，如果需要使用 TSF 中容器相关功能需要授权。
QcloudVPCReadOnlyAccess	可选	如果需要读取集群 VPC 等信息需要授权。

如担心策略范围过大，可参考访问管理的 排除故障 文档对具体接口进行授权。

其他问题

最近更新时间：2024-01-19 11:48:12

如何开通 TSF?

如果需 TSF，请先做以下准备工作：

1. 主账号在 TSF 控制台概览页 或者 CAM 控制台 创建服务角色。
2. 如果用户是子账号或者协作者，还需要主账号创建 PassRole 策略，并将策略绑定到子账号或协作者。详细操作请参考 准备工作 。

start.sh 的作用是什么?

Service Mesh 应用部署使用 tar.gz 、zip 压缩格式的程序包，在压缩包中除了应用程序外，还需要包含了 start.sh、stop.sh、cmdline 三个文件：

- start.sh：用于启动应用程序。
- stop.sh：用于停止应用程序。
- cmdline：用于检查应用进程是否存在，**没有 `.sh` 后缀。**

更多关于 start.sh、stop.sh、cmdline 的编写规范请参考 程序包格式说明 。

TSF 能单独输出部署到公司内网吗?

可以，请 提交工单 ，会有专门的技术支持人员联系您。

JVM 内存不足如何处理?

JVM 内存不足（java.lang.OutOfMemoryError）时，需要您手动设置内存大小，具体设置方法请在互联网中搜索“JVM 内存设置”。

TSF 支持哪些编程语言?

- 若使用 Spring 接入，则需使用 Java 语言。
- 若通过 Mesh 接入，则不限制编程语言。TSF Mesh 支持不同语言实现的微服务之间相互调用，并支持不同框架如 Mesh 和 Spring Cloud 运行微服务相互调用。

TSF Severless 框架使用 Node Express 构建项目，如何连接 MySQL 数据库?

TSF Serverless 默认不带 MySQL 环境的，您可以使用 腾讯云数据库 MySQL 进行操作：

1. 在本地打包环境

```
npm i -D mysql
npm i -D node-mysql
```

2. 连接 MySQL 数据库

```
const mysql = require('mysql');

const client = (sql, arg, callback) => {
  //1.创建连接
  let config = mysql.createConnection({
 host: 'localhost',
 user: 'root',
```

```
 password: 'root',
 database: 'vue'
  })
  //2.开始连接
  config.connect()
  //3.对数据库进行增删查改
  config.query(sql, arg, (err, data) => {
 callback && callback(err, data)
  })
  //4.关闭数据库
  config.end()
}

module.exports = client;
```