全流量分析平台基础运维手册

文档创建： **2023-07-11**

目录

1、组件配置文件和日志查询目录

Parser（解析器）组件：

日志文件路径：/var/log/cscmc/engine/parser/

配置文件路径：/etc/cscmc/engine/parser.cfg

Analyzer（分析器）组件：

日志文件路径：/var/log/cscmc/engine/analyzer/

配置文件路径：/etc/cscmc/engine/analyzer.cfg

Csidb（数据库）组件：

csidb 数据库配置文件：/opt/csidb/dataservice/registry.conf

csidb WEB 服务配置文件：/opt/csidb/webservice/registry.conf

采集控制器组件：

日志文件路径：/var/log/cap-controller

配置文件路径：/etc/cap-controller/cfg.ini

应用中心组件：

日志文件路径：

cta-alarm-boot 服务日志：/var/log/cscmc/app/java/cta-alarm-boot/

cta-auth-boot 服务日志：/var/log/cscmc/app/java/cta-auth-boot/

cta-business-major-boot 服务日志

/var/log/cscmc/app/java/cta-business-major-boot/

cta-node-manage-boot 服务日志

/var/log/cscmc/app/java/cta-node-manage-boot/

kafka 日志：/usr/local/bin/cscmc/app/kafka/logs/

clickhouse 日志：/var/log/cscmc/app/clickhouse-server

mysql 日志：/var/log/cscmc/app/mysql

nginx 日志：/var/log/nginx/

配置文件路径：

kafka：/etc/cscmc/app/kafka/config/server.properties

nginx: /etc/nginx/nginx.conf

clickhouse: /etc/clickhouse-server/

mysql: /etc/cscmc/app/mysql/my.cnf

服务组件： /usr/local/bin/cscmc/app/java/

2、组件配置参数说明

2.1 组件与功能关系：

组件名称	组件功能
agent	采集探针
cap-controller	采集探针控制器
parser	流量解析
analyzer	流量分析
cta-data-adapt	应用中心后台接口
nginx	Web 访问请求管理
clickhouse	流量分析结果存储
csidb	数据包存储

mysql	采集控制器存储数据、应用层数据存储
redis	Token 数据缓存

2.2 应用中心配置相关说明：

Major 组件：

#是否同步到采集控制器授权信息

license: false

#是否开启中心威胁情报功能

alarmTask: false

#本地开发去除所有定时任务

jobTask: true

#自动服务的时间延迟 分钟

autoServiceTime: 10

默认的等级是 1,即保持和资产同步一样,开启和关闭任务，如果大于 1，则是直接关闭同步

level: 1

是否使用 service-flow 时间作为 custom-app 时间

replaceAppTime: true

serviceFlow 使用资产 id 以外的指标字段,false 不使用

externalField: false

#是否开启 kafkaTask debug

kafkaTaskDebug: false

```
# 是否开启查询 serviceFlow
    serviceFlowEnable: false
# major 组件配置时间同步周期
  ntp:
      interval: 3600 #默认 1 小时同步一次,60 * 60
```

Manage 组件：

```
#查询分析器解析器 设备信息接口超时时间
 TimeSecond: 15
 job:
   jobTask: false
 appConfigSize: 1000
 mockSwitch:
   dq: false
 tcp:
   serverPort: 6000
   #解析器接受 agent 数据端口
   parserAgentPort: 6003
   #分析器端口
   analyzerPort: 6001
   #解析器端口
   parserPort: 6002
```

2.3 采集控制器配置参数说明：

配置文件路径,

env(CONFIG_FILE)config_file = /etc/cap-controller/cfg.ini;

是否启用 debug 模式运行，这会以 debug 运行 gin，同时打印 debug 日志,

env(DEBUG)debug = false;

采集器默认的 CPU 核心是数,

env(DEFAULT_AGENT_CPU_CORE)default_agent_cpu_core = 0;

采集器默认的内存使用量,单位 Byte,

env(DEFAULT_AGENT_MEM_LIMIT)default_agent_mem_limit = 0;

禁用控制器时间同步,

env(DISABLE_TIME_SERVER)disable_time_server = false;

策略过滤规则解析云资产信息的重试次数，这通常再控制器启动时使用,

env(LOAD_STRATEGY_ASSET_COUNT)load_strategy_asset_count = 10;

云平台定时同步端口信息的最小时间间隔,

env(MIN_SYNC_PORT_INTERVAL)min_sync_port_interval = 10s;

云平台定时拉取资源的时间间隔,

env(PLATFORM_RESOURCE_PULL_INTERVAL)platform_resource_pull_interval =

2m0s;

控制器 token 过期时间,

env(TOKEN_EXPIRE_TIME)token_expire_time = 2h0m0s;

agent 流量接收器过滤模式，0-不过滤，1-全过滤,

env(TRAFFIC_FILTER_MODE)traffic_filter_mode = 0;

告警日志存放时间,

env(WARNING_SAVE_TIME)warning_save_time = 24h0m0s[database];

数据库服务器 IP 地址,

env(DATABASE_DB_IPADDRESS)db_ipaddress = 127.0.0.1;

最大连接周期,

env(DATABASE_DB_MAX_CONN_TIMEOUT)db_max_conn_timeout = 1m40s;

闲置连接数,

env(DATABASE_DB_MAX_IDLE_CONN)db_max_idle_conn = 20;

最大连接数,

env(DATABASE_DB_MAX_OPEN_CONN)db_max_open_conn = 100;

数据库名称,

env(DATABASE_DB_NAME)db_name = cta_controller;

数据库密码,

env(DATABASE_DB_PASSWORD)db_password

数据库服务器端口,

env(DATABASE_DB_PORT)db_port = 3306;

时区,

env(DATABASE_DB_TIMEZONE)db_timezone = Asia/Shanghai;

数据库类型 mysql, 目前只支持 mysql,

env(DATABASE_DB_TYPE)db_type = mysql;

日志文件路径,

env(LOG_LOG_FILE)log_file = /var/log/cap-controller/cap-controller.log; 日志

同时输出到标准输出,

env(LOG_LOG_TO_STDOUT)log_to_stdout = false;

日志轮动后保留的历史文件个数，超过该配置的历史日志将会被删除,

env(LOG_ROTATE_FILE)rotate_file = 10;

日志轮动的触发大小,单位为 MByte，当日志文件的大小超过这个值时，触发一次轮动,

env(LOG_ROTATE_SIZE)rotate_size = 100[server];

控制器 grpc 消息处理超时时间,

env(SERVER_GRPC_MESSENGER_TIMEOUT)grpc_messenger_timeout = 15s;

控制器 grpc 服务器监听的端口,

env(SERVER_GRPC_SERVER_PORT)grpc_server_port = 0.0.0.0:50051;

http 服务器监听的端口,

env(SERVER_HTTP_PORT)http_port = 8081;

http 客户端请求超时,

env(SERVER_HTTPCLIENT_TIMEOUT)httpclient_timeout = 3m0s;

是否使用 https,

env(SERVER_HTTPS_ENABLE)https_enable = true;

https 服务器监听的端口,

env(SERVER_HTTPS_PORT)https_port = 9443

2.4 探针组件配置参数说明：

[default];

monitor 与 agent 通信本地的 http 端口,

env(AGENT_DAEMON_PORT)agent_daemon_port = 11117;

配置文件路径,

env(CONFIG_FILE)config_file = /etc/cap-agent/cfg.ini;

是否开启 debug,

env(DEBUG)debug = false;

是否启用加密，等号两边一定要有空格不然安装脚本处理会有问题,

env(ENCRYPT_ON)encrypt_on = false;

日志轮动后保留的历史文件个数，超过该配置的历史日志将会被删除,

env(LOG_ROTATE_FILE)log_rotate_file = 10;

日志轮动的触发大小,单位为 MByte，当日志文件的大小超过这个值时，触发一次轮动,

env(LOG_ROTATE_SIZE)log_rotate_size = 20;

日志同时输出到标准输出,

env(LOG_TO_STDOUT)log_to_stdout = false;

rpc 服务器的地址，就是控制器程序部署的服务器的地址，多个地址用,分隔,

env(RPC_SERVER_ADDRESS)rpc_server_address = 10.16.16.45:50051

[agent];

网卡描述信息过滤列表，在 windows 中适用,

env(AGENT_CAPTURE_DESC_PREFIX)capture_desc_prefix = ;

存放 agent 状态的文件路径,

env(AGENT_COMPUTER_FLAG_FILE)computer_flag_file =

/var/lib/colasoft/computer_flag.ini;

标识此 agent 采集的设备类型，分为：宿主机(pm)、虚拟机(vm)、管理类虚机(mgmtvm)、

管理类物理机(mgmtpm)、PC 终端(pc),

env(AGENT_COMPUTER_TYPE)computer_type = pm;

终端分析时，存在 CSEta 相关数据的目录,

env(AGENT_CSETA_DIR)cseta_dir = /etc/cap-agent/cseta;

禁用采集器 Daemon, env(AGENT_DISABLE_DAEMON)

disable_daemon = false;

启用 ebpf 采集功能, env(AGENT_ENABLE_EBPF)

enable_ebpf = false;

日志文件路径,

env(AGENT_LOG_FILE)log_file = /var/log/cap-agent/cap-agent.log;

是否开启网卡混杂模式来抓包,

env(AGENT_PROMISC_MODE)promisc_mode = false;

采集口分配 ringBuffer 的大小，单位为 Mb，0 表示自适应,

env(AGENT_RING_BUFFER_BLOCK_SIZE)ring_buffer_block_size = 0

2.5 平台组件调试工具使用方式：

解析器的调试页面

查看监听端口服务已开启

访问调试页面：http://解析器 ip:60028

开启分析器配置

查看监听端口服务已开启

```
[root@localhost ~]# netstat -nltp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:6001            0.0.0.0:*               LISTEN      30740/analyzer
tcp        0      0 0.0.0.0:60018           0.0.0.0:*               LISTEN      30740/analyzer
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      1281/sshd: /usr/sbi
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1549/master
tcp        0      0 0.0.0.0:8090            0.0.0.0:*               LISTEN      15144/./csidb_webse
tcp        0      0 0.0.0.0:55556           0.0.0.0:*               LISTEN      15794/./csidb_datas
tcp6       0      0 :::22                   :::*                    LISTEN      1281/sshd: /usr/sbi
tcp6       0      0 ::1:25                  :::*                    LISTEN      1549/master
```

访问调试页面：http://分析器 ip:60018

3、平台常用配置手段

3.1 组件情况查看(组件日志查询见 1-1)

解析器状态查看：

或者登录解析器服务，使用如下命令查看：

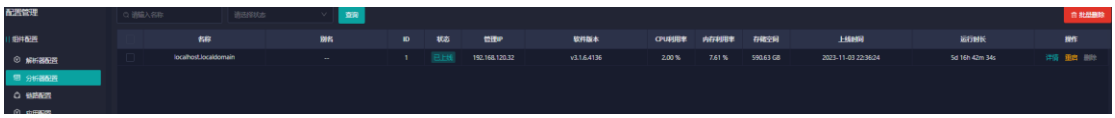
service parser status

启停解析器：

service parser start

service parser stop

分析器状态查看：



或者登录解析器服务，使用如下命令查看：

service analyzer status

启停分析器：

service analyzer start

service analyzer stop

应用中心状态查看：

systemctl status cta-alarm-boot.service

systemctl status cta-auth-boot.service

systemctl status cta-business-major-boot.service

systemctl status cta-node-manage-boot.service

采集控制器和探针状态查看：

采集控制器：systemctl status cap-controller.service

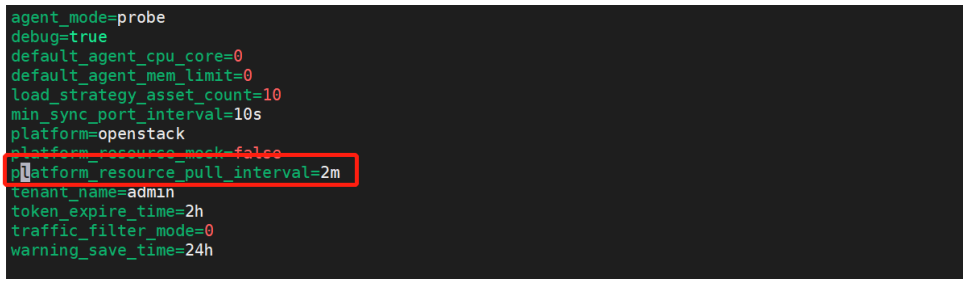
探针：systemctl status cap-agent.service

3.2 修改平台时间同步设置

登录平台之后，管理中可以修改平台时间同步相关设置

3.3 云平台同步时间间隔修改

登录采集控制器组件服务器，修改 controller 配置：vim /etc/cap-controller/cfg.ini ，将拉取资产间隔时间（platform_resource_pull_interval）改大



3.4 磁盘空间利用率配置：

登录 csidb（解析器，分析器）对应节点服务器，在 web 界面访问一下地址：

http://解析器 ip:8090

http:/分析器 ip:8090

登录默认账号密码为：

admin

!CSNTA@l23E8

然后在命令框中输入一下命令实现存储空间分配：

show synpartition （获取同步空间名称）

alter synpartition cmcstats size（空间大小，单位 G）

3.5 DPDK 性能调优手段（仅供参考，尽量在厂商陪同下操作）：

调整 dpdk-cpu 绑定

要确保使用同一 socket 上的 cpu 核心，这样抓包性能才比较好。

1） 查询网卡在 socket0 还是 socket1 上

有两种方式可以查看，默认推荐使用方式一，如果方式一执行失败，请改用方式二。

再查询前一定要确保 parser 服务已停止运行，并且无其他的 dpdk 抓包进程在运行：

如果 parser 处于运行状态，请停止：service parser stop

如果有其他的 dpdk 抓包进程请杀掉：查询进程号：ps -ef |grep dpdkcapture |grep -v grep ，如果有进行则运行 kill -9 <pid>进行杀掉

a) 方式一

进入/usr/local/bin/cscmc/engine/dpdksdk 目录，执行./dpdkenvcheck

```
[root@master dpdksdk]# ./dpdkenvcheck
EAL: Detected 48 lcore(s)
EAL: PCI device 0000:3b:00.0 on NUMA socket 0
EAL:   probe driver: 8086:10fb net_ixgbe
EAL: PCI device 0000:3b:00.1 on NUMA socket 0
EAL:   probe driver: 8086:10fb net_ixgbe
EAL: PCI device 0000:3c:00.0 on NUMA socket 0
EAL:   probe driver: 8086:10fb net_ixgbe
EAL: PCI device 0000:3c:00.1 on NUMA socket 0
EAL:   probe driver: 8086:10fb net_ixgbe
```

在默认未开启 numa 的情况下，socket -1 就表示 socket 1

```
[root@colasoft dpdksdk]# ./dpdkenvcheck
EAL: Detected 64 lcore(s)
EAL: PCI device 0000:86:00.0 on NUMA socket -1
EAL:   probe driver: 8086:10fb net_ixgbe
EAL: PCI device 0000:86:00.1 on NUMA socket -1
EAL:   probe driver: 8086:10fb net_ixgbe
EAL: PCI device 0000:af:00.0 on NUMA socket -1
EAL:   probe driver: 8086:10fb net_ixgbe
EAL: PCI device 0000:af:00.1 on NUMA socket -1
EAL:   probe driver: 8086:10fb net_ixgbe
```

b) 方式二

cat /var/log/messages |grep PCI |grep -E '0b:00.0|1b:00.0' | grep EAL

0b:00.0|1b:00.0 ，这个是步骤**错误!未找到引用源。**中获取的 PCI 信息

```
Network devices using kernel driver
===================================
0000:0b:00.0 'VMXNET3 Ethernet Controller 07b0' if=ens192 drv=vmxnet3 unused=igb_uio,uio_pci_generic *Acti
0000:1b:00.0 'VMXNET3 Ethernet Controller 07b0' if=ens256 drv=vmxnet3 unused=igb_uio,uio_pci_generic

No 'Crypto' devices detected
```

也可以找到对应的 socket 信息

2） 确定不同 socket 上的 cpu 核心

进入/usr/local/bin/cscmc/engine/dpdksdk/tools 目录，执行./cpu_layout.py

```
[root@colasoft dpdksdk]# ./tools/cpu_layout.py
============================================================
Core and Socket Information (as reported by '/sys/devices/system/cpu')
============================================================

cores =  [0, 7, 1, 6, 2, 5, 3, 4, 8, 15, 9, 14, 10, 13, 11, 12]
sockets =  [0, 1]

        Socket 0        Socket 1
        --------        --------
Core 0  [0, 32]         [1, 33]
Core 7  [2, 34]         [3, 35]
Core 1  [4, 36]         [5, 37]
Core 6  [6, 38]         [7, 39]
Core 2  [8, 40]         [9, 41]
Core 5  [10, 42]        [11, 43]
Core 3  [12, 44]        [13, 45]
Core 4  [14, 46]        [15, 47]
Core 8  [16, 48]        [17, 49]
Core 15 [18, 50]        [19, 51]
Core 9  [20, 52]        [21, 53]
Core 14 [22, 54]        [23, 55]
Core 10 [24, 56]        [25, 57]
Core 13 [26, 58]        [27, 59]
Core 11 [28, 60]        [29, 61]
Core 12 [30, 62]        [31, 63]
```

3） 修改/etc/cscmc/engine/parser.cfg 配置，调整完成后保存配置

如果网卡在 socket 1 上，则配置 socket 1 的 cpu 核心，如果网卡在 socket 0 上，则配置 socket 0 的 cpu 核心，dpdk 绑定的 cpu 核心为独占，不能被其他使用

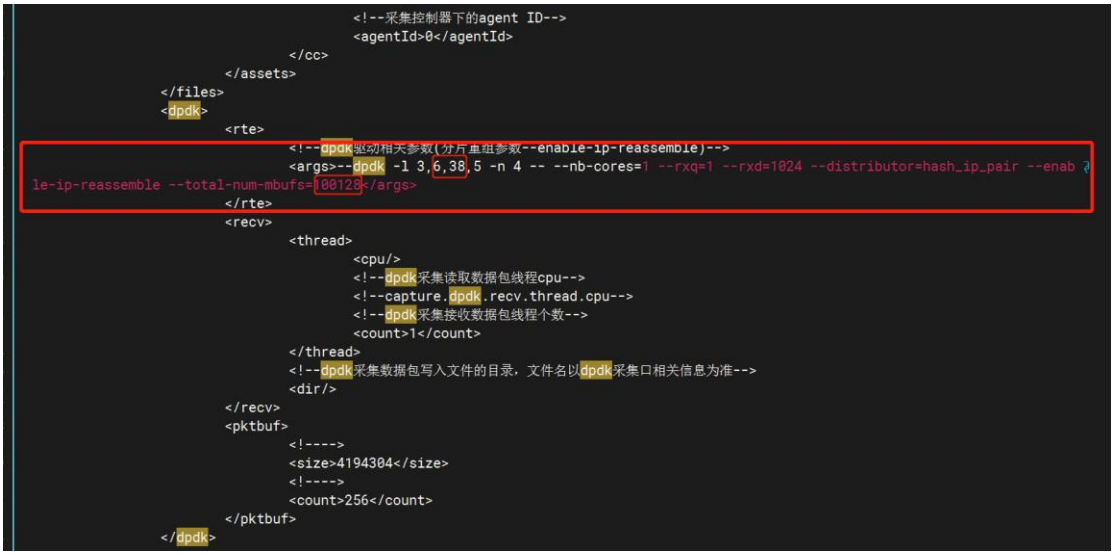
以 socket 0 为列，我们使用核心 6 进行抓包，需要排除对端核心 38 的配置

修改前：

<args>--dpdk -l 3,4,5 -n 4 -- --nb-cores=1 --rxq=1 --rxd=1024 --distributor=hash_ip_pair --enable-ip-reassemble --total-num-mbufs=10012</args>

修改后：<args>--dpdk -l 3,6,38,39 -n 4 -- --nb-cores=1 --rxq=1 --rxd=1024 --distributor=hash_ip_pair --enable-ip-reassemble --total-num-mbufs=100128</args>

将 4 修改为 Core6 的核心信息，并且修改缓存包数--total-num-mbufs 为：100128



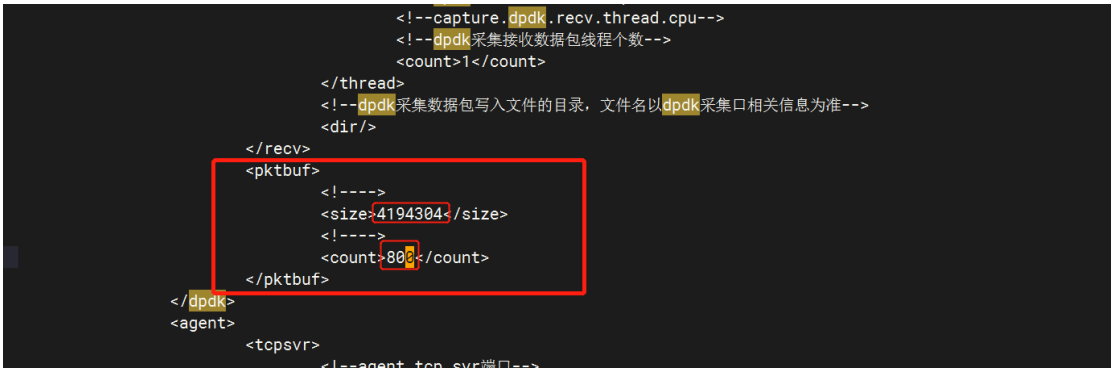
调整 dpdk-buffer 配置

调整程序使用的大页配置，调整完成后保存配置

1） 修改/etc/cscmc/engine/parser.cfg 配置中的 size 和 count 参数

size 参数：size 修改为**错误!未找到引用源。**章节中大页设置，例如：4G，则 size 的值为(4*1024*1024)：4194304；例如 12G，则 size 的值为：(12*1024*1024)：12582912

count 参数： 例如 4G，count 配置 800（配置参考原理：count 大小 * 4MB 小于大页配置，例如： 800*4MB <4G）



4、平台组件常见异常排障手段

4.1 各组件异常排障简介

中间件服务异常排查：
1、使用 journalctl 命令排查服务启动失败原因，journalctl -xe | grep 服务名称，如：journalctl -xe | grep mysql
2、查看相关服务日志，服务日志路径见：组件的日志和配置文件路径
应用中心服务异常排查：
1、查看相关服务日志，服务日志路径见：
2、停止服务后，手动运行服务，如图：组件的日志和配置文件路径
解析器分析器服务异常排查：
1、查看相关服务日志，服务日志路径见：组件的日志和配置文件路径
2、停止服务后，手动运行服务，解析器手动运行方法：
/usr/local/bin/cscmc/engine/parser -d ，
分析器手动运行方法：
/usr/local/bin/cscmc/engine/analyzer -d

4.2 Clickhouse（数据库）启动失败常见问题

首先运行： ldd /usr/bin/clickhouse-server
查看是否有出现 xxx not found ，如果有的话，则表示操作系统缺少相关的依赖库，则需要上传依赖库后重启 clickhouse 服务
如果第一步的依赖库没有问题，则查看错误日志： cat /var/log/clickhouse-server/clickhouse-server.err.log，查看最新时间的日志信息
如果日志报错：Listen [::] failed
A：这个是由于默认配置是基于网卡支持 ipv4 和 ipv6 两种模式，如果网卡只支持了 ipv4（通过命名： ip a 查看网卡是否有分配 ipv6 地址，如果没有则表示只支持了 ipv4），则需要将修改配置文件
vim /etc/clickhouse-server/config.xml
将<listen_host>::</listen_host>修改为： <listen_host>0.0.0.0</listen_host>
然后再重启下 ck 服务： systemctl restart clickhouse-server

4.3 平台下载数据包失败

登录解析器后台，查看/colasoft-cscmc-data/log/目录下的日志文件，确认 csidb 状态是否，命令：cat dataservice* | grep “start db”
检查三台解析器存储空间是否正常：df -Th
分别登陆解析器的 csidb 的页面：http://解析器 IP:8090，检查存储分区大小：show synpartition
分别登陆解析器的后台，监控 parser 日志：tail -f /var/log/cscmc/engine/parser/parser.parser.* -n 1，然后在页面上进行数据包下载，查看日志监控中下载数据包的 sql 语句
如果不能下载数据包，将下载数据包的 sql，分别放在解析器的 csidb 页面，进行查询，看是否有数据返回

4.4 对接集群显示异常链接

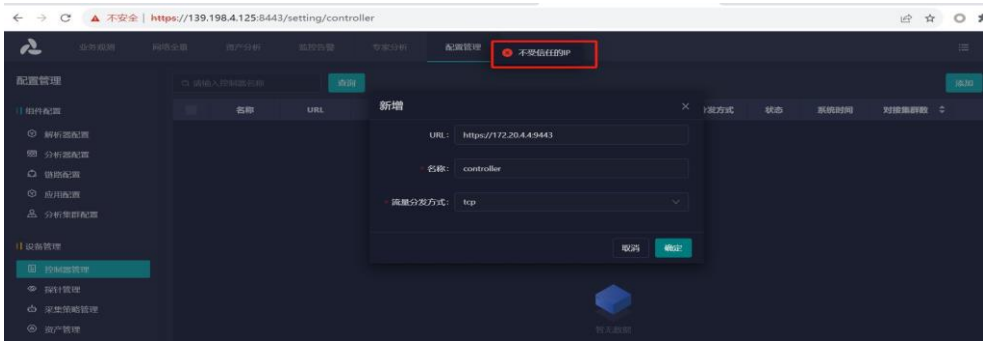
拉取集群资产时，集群返回数据比较慢，然后我们默认是 2 分钟拉取一次，相对比较频繁，所以出现一会已连接，一会连接中

调整方法，修改 controller 配置：vim /etc/cap-controller/cfg.ini ，将拉取资产间隔时间（platform_resource_pull_interval）改大

```
agent_mode=probe
debug=true
default_agent_cpu_core=0
default_agent_mem_limit=0
load_strategy_asset_count=10
min_sync_port_interval=10s
platform=openstack
platform_resource_mock=false
platform_resource_pull_interval=2m
tenant_name=admin
token_expire_time=2h
traffic_filter_mode=0
warning_save_time=24h
```

配置修改完成后，重启 controller 服务：service cap-controller restart

4.5 添加多个控制器，提示 IP 不被信任异常



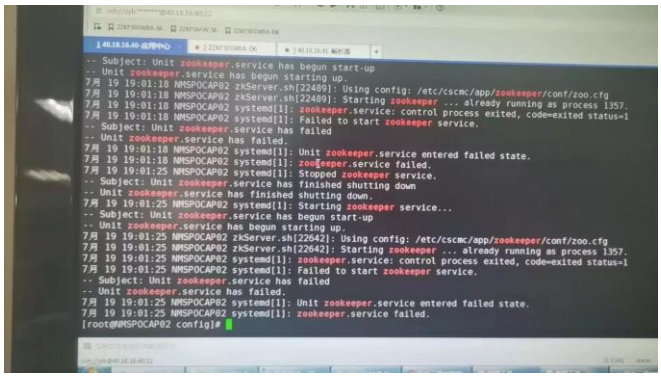
处理方法：

1、编辑/usr/local/bin/cscmc/app/java/ControllerIp.txt 文件，将 IP 添加到文件中，文件不存在则手动创建

2、重启 business 服务：service cta-business-major-boot restart

4.6 Zookeeper 服务启动失败

查看 zookeeper 错误原因，报进程已存在，但是该进程不是 zookeeper 进程



解决方法：rm -rf /colasoft-cscmc-app/zookeeper/data/zookeeper_server.pid

再重启 zookeeper 服务：service zookeeper restart

4.7 cta-node-manage-boot（应用中心）服务启动失败

在线解码库初始失败

1、移除在线解码 so 库：mv /usr/lib64/liboldparser.so /home

2、编辑 cta-node 服务，关闭在线解码 so 初始化，vim /etc/systemd/system/cta-node-manage-boot.service，加上-Dcta.so.init=false