[SCA] 网络质量模块优化设计

[Title]{Technical Project Name} Technical Solution

Revision history

Version	Revision date	Revisor	Revision content
v1	*2022.02.22*	*xxx*	*xxx*
v2	*2022.03.15*	*xxx*	*xxx*

Ownership

Product Manager	*The PIC of Product manager*
Project Manager	*The PIC of Project Manager*
Native Dev	*The PIC of Android or iOS Dev*
RN Dev	*The PIC of React native dev*
Server Dev	*The PIC of non-native dev*

Designer	*The PIC of designer*
QA	*The PIC of QA*

Resources

PRD	*The link of PRD(s), created by product manager*
Figma	*The link of UI design*
Transify	*The link of transify in https://transify.seagroup.com*
Git Repo(Optional)	*The git repo or branch for this project if necessary to indicate*
Dependent Service Doc(Optional)	*SDK integration doc, official guideline, etc.*
Project Schedule Page (Optional)	*Usually created by project manager, timeline of the document*
Feasibility Study Doc (Optional)	*Usually feasibility study is ahead of PRD.*

Summary

Background

当前的网络诊断工具，有着以下缺陷：一是诊断结果不够直观、准确、全面，二是入口界面未经过PM和PD的设计，无法直接产品化。该模块需要进一步优化。

Goal

1.校准当前诊断功能的结果。

2.优化结果，给出更完善的信息，以及简单的诊断结论。

3.新增实时监测的相关功能，及对外接口。

4.优化对外接口、新增一些配置和数据库实现。

Abbreviations (optional)

Here to describe the related abbreviations, eg.

ABB	Full name	Description
TTL	Time To Live	数据包被路由器丢弃之前允许通过的网段数量。
RTT	round-trip time	网络请求从起点到目的地然后再回到起点所花费的时长。
MTU	Maximum Transmission Unit	最大传输单元，限制的是数据链路层的payload，也就是上层协议的大小，例如IP，ICMP等。

Overall Design

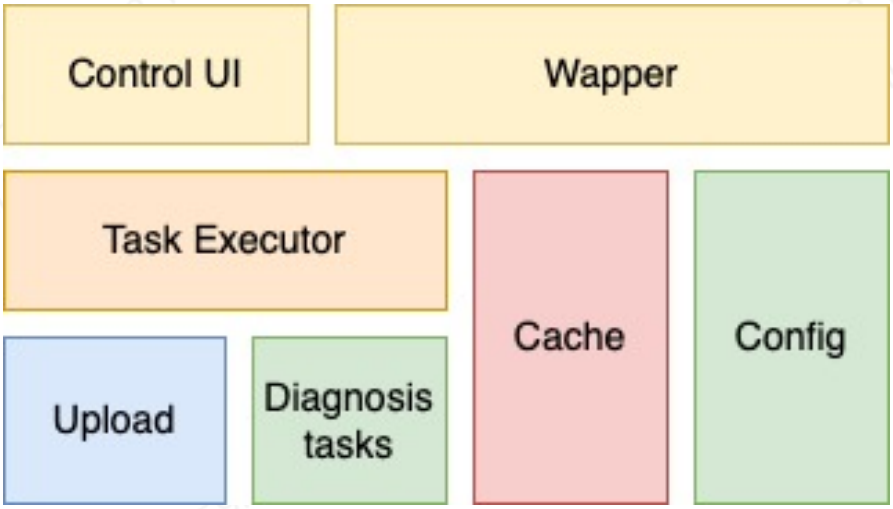
Improvement(revamp/refactor) for existing feature

- Current

原有方案设计文档：[SCA] 网络诊断技术方案设计

诊断收集的信息可包含如下信息：本机设备的基本信息（Device），基本网络详细信息（Net），网络通畅性和稳定性（Ping），Http请求检查（Http），检查本机Host（Host），检查常见端口（Port Scan），检查所经路由情况（TraceRoute），检查所经路由的传输单元（Mtu Scan），检查DNS域名解析（NSLookup）。

软件结构：

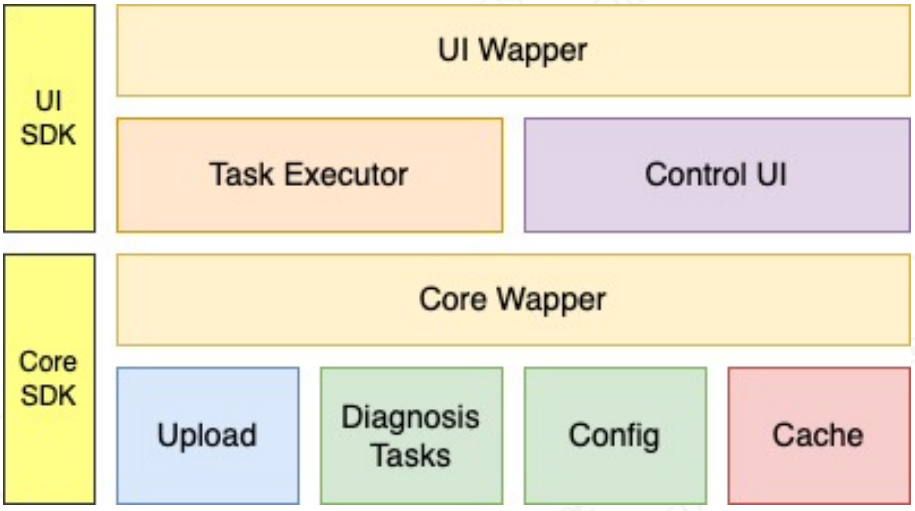


外层主要把操作页面提供给业务方，其他功能的调用不够友好，如设置配置信息、开启诊断任务等，并且未提供结果上报的定制化，默认上报到QMS。

- Improvement

在保留原有诊断项的基础上，检查信息的准确性，如traceRoute结果的rtt。对于Ping检查，需要加入外网服务（如Google网站）和业务服务的对比，给出连通性的结论；DNS解析给出结论，是否正常解析；TraceRoute给出更多信息，检查节点数量、可获取IP节点数、rtt未超时节点数；Http检查新增实际下载速度的测量，下载小文件。

新增提供实时网速监控的监听回调，实时网速计算方案见附加内容。



改善包装模块，整合对外提供的功能。下沉基础功能为Core基础库，包含具体诊断任务、上传功能、数据缓存和配置，外部依赖Core库可以自行整合具体诊断任务的执行，或者加入自定义的任务配置自定义的数据；已有的UI和完整的诊断任务调度，整合到UI库，WMS已集成的功能可以依赖该库，该库自动依赖Core库，保留对WMS已集成功能的兼容。

改善诊断流程，分阶段逐步诊断；丰富配置信息。另外，DB存储要废弃GreenDao，改用内部存储库。

Detailed Design

Improvement(revamp/refactor) for existing feature

1 诊断功能的优化

1.1 网络状态信息

网络基本信息包含：网络是否可用、是否连接WI-FI、移动网络信息（网络类型、信号强度等）、WIFI信息（Wi-Fi名称、信号强度等）、localHost（包括IPv4和IPv6）。

主要都通过Android系统API能力获取，见原有设计方案，可以保留。涉及电话权限和定位权限，缺失权限可能导致部分结果缺失，业务方自行选择是否申请权限。

1.2 NSLookup

主要检查DNS域名解析，用不同解析策略对目标服务器的域名进行解析。

只关注默认策略解析DNS是否成功，作为新增的总体结论。

1.3 Ping网络连通性检查

可以利用控制台指令，运行/system/bin/ping的Ping程序，以实现Ping。除了指明特定服务器的Host，还可以指定Ping的次数（发送包数量），以及超时时间。次数为十次，超时时间半分钟。

从输出流中解析结果，可以得到目标服务器的IP地址，发送包、接收包、丢包率、TTL生存时间、最大RTT、最小RTT以及平均RTT等。

执行的默认命令为，/system/bin/ping -c 10 -w 32 [host]。

丢包率0判断为连通性良好，小于50%判断有波动，大于50%判断为弱网，100%丢包判断为不连通，新增判断结果描述。

新增一个新的Ping任务，去检查与公网的连通性。默认情况下可以用Google网站，也会开放配置项供业务方配置公网host。

1.4 Http下载检测

主要检查对目标服务器的实际下载速度，Http可以使用OkHttp来实现。

新增检查实际下载速度测量，下载服务器的固定文件（开放配置下载地址，不同的业务方自行配置），下载时间有个上限，可以外部配置，默认最多下载5s，实际下载速度为下载字节数/下载时间。

1.5 MTU扫描

MTU扫描是扫描与目标服务器之间所有的路由，其最大传输单元的最小值。使用Ping命令实现，扫描的范围按照数组{1500, 1492, 1472, 1468, 1430, 1400, 576}执行。

执行的默认命令为，/system/bin/ping -c 1 -w 1 -s [mtu scan array] [host]。

1.6 TraceRoute

Trace Route可以靠Ping命令模拟，对比了两种模拟方式，可以得到相同的结果。

新增统计检查的总节点数，可获取IP节点数和可获取rtt节点数。

新增结论，总节点数大于30且hop对应30的节点无法获取IP，可以判断服务器不可达；其他情况可以认为可达。

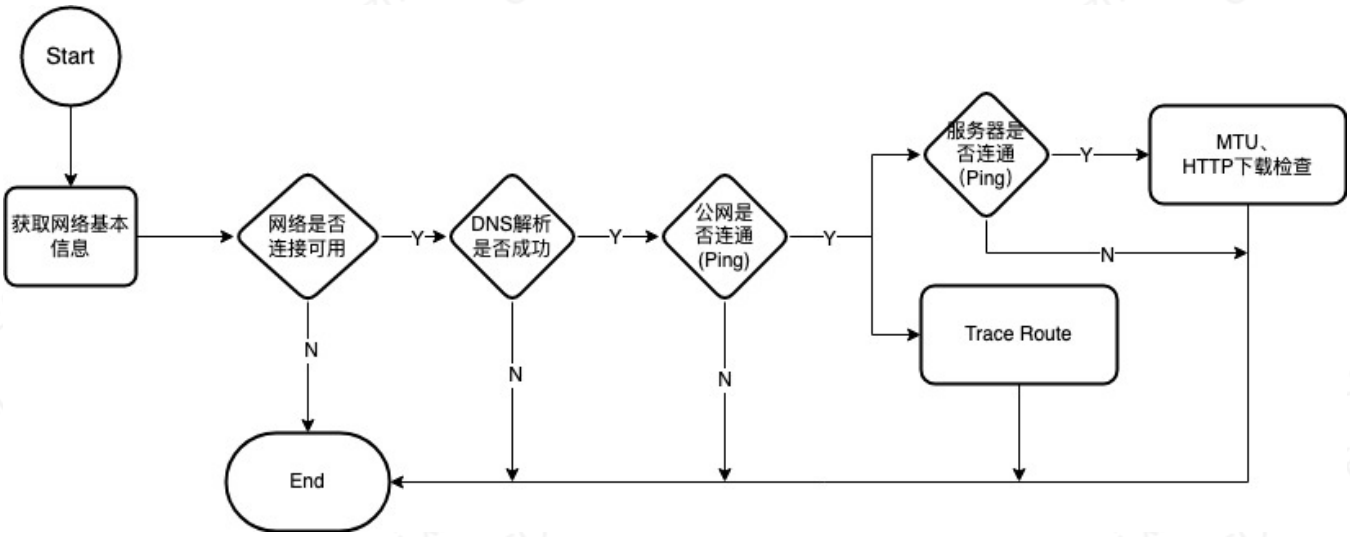
执行的默认命令为，扫描节点：/system/bin/ping -c 1 -w 5 -t [hop] [host]，然后Ping对应节点：/system/bin/ping -c 3 -w 5 [hop ip]。

1.7 Realtime Speed

实时网速检测。提供开始、结束采样的方法，外部可以自行决定调用时机，结束采样时返回采样结果。也可以传入采样时间，执行process方法，自行采样一次，并返回结果。

2 诊断流程优化

原有诊断流程为默认全项目并行执行，实际上可以根据初步诊断省略掉不需要的诊断项。



3 数据上报

可以沿用原有的上报策略及协议。上报新增上报API的配置。

诊断结束则尝试上报，上报成功则删除本地数据库数据；另有一个轮询任务检查数据库是否有未上报成功的数据，有则上报，上报成功则删除这些数据。

Interface Design

1 上报协议

默认上报API：

https://autobahn.ssc.test.shopeemobile.com/data/
https://autobahn.ssc.shopeemobile.com/data/

post请求的body字段见下表：

字段名称	字段类型	默认值	含义
bt	String	空	项目名，外部配置
sbt	String	空	子项目名，外部配置，未配置则默认与bt相同
env	String	test	网络环境，如test、uat、live等
dt	String	code_log	固定写死，区分上报的类型
uid	String	空	外部配置
ut	String	空	外部配置，原来用于上报仓库名
sid	String	设备ID	设备ID，安全系统设置读取Settings.Secure.ANDROID_ID
r	int	10	暂无含义
s	String	/a/b	暂无含义
a	String	空	外部配置，原有用于上报国家信息
pf	String	android	固定，用于区分平台
d	Object		详细数据data

data字段：

字段名称	字段类型	默认值	含义
ct	String		当前时间戳字符串，单位秒

type	String	verbose	暂无含义
tid	String	当前线程ID	Process.myTid()
pid	String	当前进程ID	Process.myPid()
tname	String	当前线程名称	Thread.currentThread().getName()
module	String	net diagnosis	固定
log	String		诊断结果上报的正文，这里上报的是json字符串

原有的上报正文字段参考了数据库的字段，可以保留。另外增加了一项基础信息的结构对象：

字段名称	字段类型	默认值	含义
timeoutUrl	String	空	超时触发自动诊断的url
modifyTimeString	String	空	诊断时间字符串
modifyTime	long	0	诊断时间戳
isManual	int	1	是否为用户手动触发，用于前端告警过滤数据，只有自动触发才会告警
host	String	空	诊断的服务器Host
wareHouse	String	空	仓库，业务相关数据

示例：

```
"Base":{
        "timeoutUrl":"/api/v2/pda/config/pdanotification/get_notification_sound_list",
        "modifyTimeString":"2021-11-24 16:36:26",
        "modifyTime":1637742986092,
        "isManual":0,
        "host":"http://wms.ssc.shopee.com.my",
        "wareHouse":"MYL"
}
```

Base数据与前端告警系统有依赖，所以需要保留，也可加入自定义数据，由外部配置加入。

2 配置协议

配置参数类：DiagnosisConfigInfo，其原有变量：

变量名称	变量类型	默认值	含义
mProject	String	空	项目名称，对应上报字段bt和sbt
mEnv	String	test	网络环境，对应上报字段env
mUid	String	空	用户ID，对应上报字段uid
mCountry	String	空	国家，对应上报字段a
mWareHouse	String	空	仓库，对应上报字段ut
mBackEnable	bool	true	后台自动触发诊断是否开启
mBackDiagnosisTimeInterval	long	60 * 60 * 1000	自动触发的最小时间间隔，单位毫秒
mBackTimeoutInterval	long	5 * 1000	触发自动诊断的业务接口超时时间，单位毫秒
mBackUploadTimeInterval	int	15	滞留数据尝试上报的时间间隔
mBackUploadTimeUnit	TimeUnit	TimeUnit.MINUTES	滞留数据尝试上报的时间间隔单位

新增字段：

变量名称	变量类型	默认值	含义
mUploadUrl	String	https://autobahn.ssc.test.shopeemobile.com/data/ https://autobahn.ssc.shopeemobile.com/data/	上报API

mPublicHost	String	www.google.com	权威公网host，用于对照
mDownloadUrl	String	空	服务器小文件下载地址，用于测量下载速度
mDownloadMaxTime	int	5000	最大下载时间，单位ms
mSubProject	String	空	子项目名称，对应上报字段sbt

3 对外接口优化

对外功能接口类 NetDiagnosis，已提供的功能保留，将提供于diagnosis-ui库。

基础库提供TaskChain类，来控制多个任务的执行，提供了简单的任务链和节点的操作方法。

可以设置监听任务链过程：

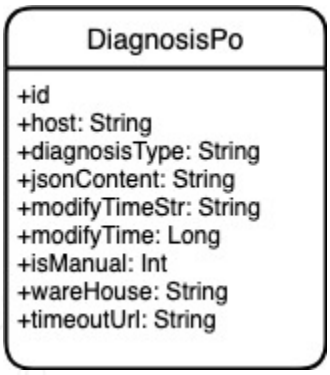
```
/**
 *
 */
public interface ITaskChainCallback {
    /**
     * key
     *
     * @param taskKey
     */
    void onTaskStart(String taskKey);
    /**
     *
     *
     * @param data
     */
    void onTaskFinish(DiagnosisPo data);

    /**
     *
     *
     * @param result
     */
    void onAllFinished(List<DiagnosisPo> result);
}
```

每个Task都可以设置单独的监听：

```
/**
 *
 *
 * @param <T>
 */
public interface ITaskCallback<T> {
    /**
     *
     * @param data
     */
    void onTaskFinish(T data);
}
```

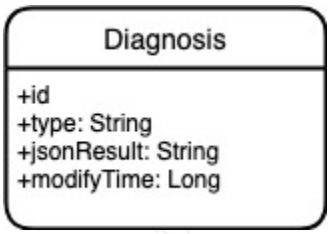
Storage Design (Optional)

原有的数据库字段：



其中包含了一些重复字段如修改时间、超时URL等，以及一些业务相关的字段WareHouse。新的数据库设计可以考虑去除这些。

不同的诊断项对应不同的Json字符串结果，新的type可以用于区分不同的诊断类型，也可以加入自定义的类型，来记录一些基础信息和业务信息，旧数据库中去除的字段可以放到自定义内容中。保留了long类型的修改时间，是为了用于检索。



默认诊断项的字段如下，诊断任务结果的Bean字段也是参考如下字段。

1.网络基础信息

Type：network_info

字段名称	字段类型	字段含义
net_available	bool	网络是否可用
is_wifi	bool	是否连接Wi-Fi
wifi_name	String	连接的Wi-Fi名称
wifi_rssi	int	Wi-Fi信号强度，单位dBm。
mobile_net_type_value	int	移动网络的类型值，Android系统定义
mobile_net_type	String	移动网络类型，2G ~ 5G
mobile_level	int	移动信号等级，0 ~ 4
mobile_dbm	int	移动信号强度，单位dBm，-100 ~ 0
mobile_asu	int	移动信号强度，单位asu，dBm = -113 + 2 * asu
client_ip	String	本地IP
dns_server	String	本地DNS服务器IP地址
local_host	List<String>	localHost，包含IPv4和IPv6

2 DNS解析

Type：nslookup

字段名称	字段类型	字段含义
total_time	int	该诊断项总耗时

dns_server	List<String>	本地DNS服务器IP列表
result	bool	是否能解析成功，所有子项取或
lookup_list	List<LookupBean>	解析详情列表

LookupBean：

字段名称	字段类型	字段含义
method	String	默认方法"default"（InetAddress类）或者指定DNS服务器IP的方法"x.x.x.x"
time	int	解析耗时
result	bool	是否能解析成功
ip_list	List<String>	解析到的IP列表，失败则为空

3 网络连通性Ping

Type：ping

字段名称	字段类型	字段含义
total_time	int	ping结果的time总耗时
ip	String	目标的IP
address	String	目标的地址，可以是IP
loss_rate	float	丢包率
rtt_avg	float	平均rtt
rtt_mdev	float	rtt平均偏差
ttl	int	剩余ttl，可以大致判断经过的路由节点数。
conclusion	String	判断连通性的结论：流畅、弱网、不稳定、不可达。

4 MTU检测

Type：mtu

字段名称	字段类型	字段含义
total_time	int	该诊断项总耗时
mtu	int	测量到的mtu结果

5 Http检测

Type：http

字段名称	字段类型	字段含义
download_speed	String	实际下载速度
speed	float	下载速度值，单位kb/s
download_time	int	下载时间，单位ms
download_url	String	下载网址
download_size	float	下载内容的大小，单位KB

6 TraceRoute

Type：trace_route

字段名称	字段类型	字段含义
total_time	int	该诊断项总耗时
trace_route_list	List<TraceRouteBean>	traceroute的节点列表
hop_total_count	int	检查的总节点数
hop_ip_count	int	可以获取到IP的节点数
hop_time_count	int	可以获取到RTT的节点数
result	bool	结论判断是否可达

TraceRouteBean：

字段名称	字段类型	字段含义
hop	int	节点号
ip	String	节点IP，默认“*”
times	List<String>	节点的RTT列表，size对应发送的包数-c

7 RealTime Speed

Type：Realtime

字段名称	字段类型	字段含义
sample_time	int	采样时间，单位ms
total_speed	float	总的网速，单位kB/s
tx_speed	float	上行速度，单位kB/s
rx_speed	float	下行速度，单位kB/s

Data track

除了上报的结果数据，可以增加产品化相关的埋点。

Monitoring

暂不新增告警。

Appendix

实时网速监测

对比使用设备下行流量变化计算（后称方案一），和Facebook开源库network-connection-class 的使用（后称方案二）。

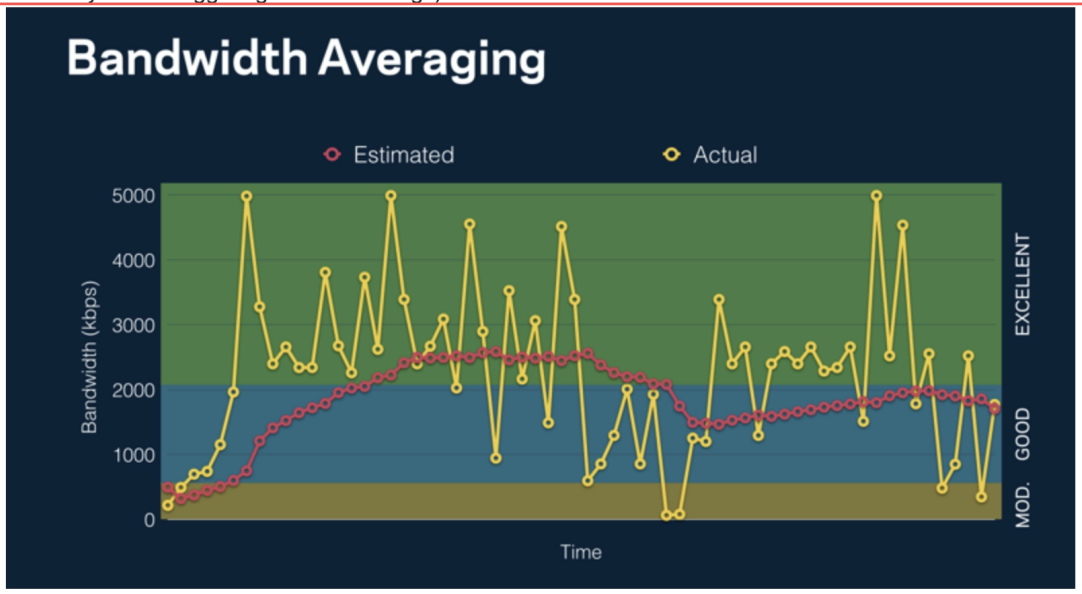
实验demo是在一个周期性的任务中，开始任务时统计时间SystemClock.elapsedRealtime()，和下行流量TrafficStats.getTotalRxBytes()，等待一秒后，计算时间差和流量差。流量差比上时间差则是方案一的结果，单位默认为KB/s；将流量差和时间差传入采样方法，ConnectionClassManager.getInstance().addBandwidth(diffBytes, diffTime)，再从ConnectionClassManager.getInstance().getDownloadKBitsPerSecond()方法获取平均带宽，单位为Kbit/s，需要除以8换算为KB/s。

从应用启动时开始，统计了十几秒的数据：https://docs.google.com/spreadsheets/d/1tfGymHBpFi2HmFa3EVRSM1bp_-T1F4l1B__1xR5-xHo/edit?usp=sharing

可以发现，方案二的平均带宽在时间拉长后不再具有实时性，对于网速实时显示的需求，方案一更合适。

方案二的网络质量评定模块，更具有价值。从官方说明可以看到，方案二故意磨平了一些峰值，并具有一定的滞后性，所以他关注的是设备的**整体网络质量变化。**

> Network Connection Class currently only measures the user's downstream bandwidth. Latency is also an important factor, but in our tests, we've found that bandwidth is a good proxy for both.
>
> The Network Connection Class library takes care of spikes using a moving average of the incoming samples, and also applies some hysteresis (both with a minimum number of samples and amount the average has to cross a boundary before triggering a bucket change):



MileStones

见[SCA] 网络质量模块优化 roadmap

Tickets

Task content	PIC	JIRA Ticket
1 sentence summary of this sub task content, can be the title of JIRA ticket	*Android dev name*	SPFE-47978 - [Android]Template Task WAITING