Released Note:

当前版本：v18\_1.2

日期：2018/11/22

Author：Klay Lin

简介：在GUI Code v18\_1.1的基础上增加SAN亚健康监控报警功能 ：

目录：

1.背景原因

2.内容

3.增改代码

4.文件配置及操作方法

5.测试状况

6.已知问题

7.需安装模块

1. 背景原因：在包含HA-AP的一个SAN架构中，SAN的亚健康状态（SAN亚健康状态指SAN链路不稳定）会导致HA-AP引擎重启，所以，我们加入了Engine reboot跟ABTS&Queue full预警。此外，Switch的健康状态也会影响到SAN是否处于亚健康状态（存在Encout或Discc3），在Encout或Discc3过多的情况下会导致引擎重启。所以，有必要加入Switch的健康监控功能，在Switch亚健康状态时发送提醒用户的警报，不会产生对HA-AP的健康状态的误报，导致HA-AP重启
2. 内容：
   1. 定时器：30分钟监测一次SAN的健康状态
   2. 判断SAN亚健康的条件(30min/time)：
      1. Engine运行时间小于30分钟
      2. Engine ABTS&Qfull达到预警值

（预警值现为0，可通过更改EngineAnalyze.ABTS\_signal改预警值）

* + 1. Switch Encout or Discc3不为0
  1. SAN亚健康状态预警机制：
     1. 邮件警报机制：包括当前造成SAN亚健康的status，产生错误信息的设备的IP

eg：HA Appliance Warning; status: ['Engine Reboot'] （email 主题）

This is HA Appliance emailing for getting help.  
 Reboot Engine IP is :['172.16.254.71'] （email 内容）

* + 1. 用户反馈机制：

当SAN处于亚健康状态，就会引发邮件预警机制，以及在网页错误信息表格记录IP。

网页有一个用户反馈“确认按钮”，用于程序检查用户是否响应错误反馈

* + - 1. 假如发送邮件后用户没能及时检查SAN的环境，没有看网页信息，没有按下“确认按钮”，即使在下一时段内没检测到SAN有亚健康状态，邮件预警还会继续发上一次检测到的错误状态及IP，网页错误信息也不会清除上一次的IP，按下确认按钮会停止邮件发送及清除错误IP（当然，用户没按按钮的时间段检测到新的错误状态会发送新邮件，显示新错误IP）
    1. 当SAN处于亚健康状态时，会在创建的历史文件中记下当前的状态 信息，可以设置保存次数（详见4.1.2）
  1. 网页展示信息：
     1. 从上往下依次以表格方式显示Engine and Switch的实时状态跟端 口数据，包括Engine的uptime等，Switch的encout，discc3等
     2. 以表格形式显示导致SAN亚健康状态的三个条件（详见2.2），当

SAN处于亚健康状态，在当前错误条件的表格记录对应IP

Eg：Engine Reboot IP（if has）

Engine ABTS&Qfull IP（if has）  
Switch Error IP（if has）

* 1. 错误历史信息展示，查看方法详见4.4

1. 增改代码：

\_\_builtin\_\_.SIGNAL:当SAN亚健康，SIGNAL[0]为1，会有邮件预警及网页预警

\_\_builtin\_\_.ErrList：存放产生Error的Engine和Switch的IP的dict

\_\_builtin\_\_.flask\_ptr: 用于表示Flask服务器是否打开和关闭的变量

\_\_builtin\_\_.Switch\_info

* 1. 定时器：

GUI main中增加function task\_30m

* 1. Switch错误信息检测：GUI code /source 下增加SWPortErrorAnalyze.py
     1. class SSHConnection：连接switch
     2. function findDataAndErr：根据switch的port number选取switch info
     3. function SWPortErrorAnalyze：分析switch info并记录到log文件之中，包括FramTX FramRX encout Discc3 LinkFL LossSC LossSG。之后创建并写入switch\_info

代码：strPortErrorFileName = 'SW\_porterrshow\_{}.log'.format(

lstSANSwitchIPs[indexEngineIP])

Switch\_info[indexEngineIP]['PortInfo'][intPortNum]= lstErrInfo

F=['FramTX','FramRX','encout','Discc3','LinkFL','LossSC','LossSG']

* + 1. function SwitchAlarm：switch预警，当'Encout','Discc3'不为0，表示 switch处于亚健康状态，Switch\_trigger=1，添加故障IP到ErrList中
    2. function ClearPortError：清除端口错误
    3. function SaveSwInfo\_hist：保存错误历史文件
  1. Engine reboot还有ABTS&Queue full检测：
     1. GUI code /source 下增加EngineAnalyze.py

function ABTS\_accumulative：判断ABTS&Queue full，大于预设值，ABTS\_trigger = 1

function Restart\_Alarm：判断引擎开机时间，如果小于30min，Alarm\_trigger = 1（Engine reboot）

function errIP：出现Engine reboot或者ABTS&Queue full，按实际情况找出引擎IP，传到ErrList中

function Fileinitial：将错误信息写入历史文件

* + 1. ParseEngineVPD.py中增加内容：
       1. function initEngineInfo添加engine\_info[i]['ABTS and queue\_full'] = {}

（把'ABTS and queue\_full添加到engine\_info）

* + - 1. line102:添加self.\_\_decodeABTS(i)，（调用写入VPD命令的函数）
      2. line207:新加function \_\_decodeABTS：解析VPD文件的端口命令

（与ABTS and queue\_full相关）

* + 1. ControlEngine.py中增加内容：

line156-160：if (self.InputToExpect( "aborts\_and\_q\_full a1\r", "CLI>", 'vpd')): return False

#将ABTS and queue\_full各端口命令写入VPD文件

* 1. 邮件报警：
     1. GUI code /source 下增加SendAlarmEmail.py文件：

function SanAllGood：添加造成SAN亚健康的条件（详见2.2）到status列表，并调用SendEmailNotification函数

function SendEmailNotification：调用ControlEmail.py中的classxEmail()的函数auto\_send\_message

* + 1. ControlEmail.py中的class xEmail()的函数auto\_send\_message添加代码：根据status列表的内容发送确切的邮件内容（邮件格式详见2.3.1）
  1. 网页预警：网页的架构采用的是Flask框架，因其具有简洁且扩展性强的优点。Flask框架在项目中的组成分别是static文件夹，templates文件夹及FlaskStartUp.py，以及在DisplayMenubar.py里增改部分代码
     1. static文件夹：存放网页引用的静态文件（CSS,JS,image）
     2. templates文件夹：存放HTML文件
     3. FlaskStartUp.py：
        1. function check\_mirror\_status：检查引擎镜像组的状态
        2. function home：与交换机以及引擎交互，获取switch\_info以及engine\_info；

用户反馈机制（有反馈，SIGNAL[0]=0）:

if request.method == "POST":

username = request.values.get('but')

if username == '1':

SIGNAL[0]= 0

* + - 1. function main：打开服务器的函数
      2. function shutdown\_server：关闭服务器的函数
    1. DisplayMenubar.py里增改代码：
       1. 在# create Development Tools menus部分代码中增加打开/关闭Flask服务的按钮

iogmenu.add\_command(label="Open Flask Server", command=self.\_\_Flasktester)

iogmenu.add\_command(label="Close Flask Server", command=self.\_\_CloseServer)

* + - 1. function \_\_Flasktester：打开服务器的线程，调用function \_\_OpenWeb
      2. function \_\_OpenWeb：打开服务器的函数，调用Fmain.py的main函数
      3. function \_\_CloseServer：关闭服务器的函数

1. 文件配置及操作方法：
   1. 增改Config.ini文件（GUI code /zip路径中）
      1. 添加两个列表（根据真实情况增改）：

eg：

[SWPorts]

172.16.254.75 = [3,1]

172.16.254.76 = [3,2]（Switch的IP及端口号）

[SWSetting]

switchname="brocade" （交换机品牌）

port = 22

username = （交换机用户名）

password = （交换机密码）

（程序设定，请将所有交换机设相同账号并写入username，password也要相同）

* + 1. 错误历史文件保存次数，可修改：

[HistoryTimes]

SwitchTimes=11 （Switch历史错误信息保存次数）

EngineTimes=10 （Engine历史错误信息保存次数）

* 1. 邮件预警收发邮箱设置：详见documents/GUI Code description v1.0CN .docx文件
  2. 如何打开关闭预警网页：
     1. 运行GUI code，点击Development Utility，点击Open Flask Server，查看命令行信息，找到Running on <http://127.0.0.1:3333/> 这行数据，复制<http://127.0.0.1:3333/> 网址，到浏览器中输入该网址
     2. 关闭Flask Server：回到GUI Code面板，点击Development Utility，点击Close Flask Server
  3. 如何查看详细错误历史信息：
     1. 直接网页打开链接：网页标题栏—-其他—-历史数据—-Engine/Switch
     2. 到GUI code/Historyfile路径下查看Switch和Engine的错误历史文件

1. 测试状况：
   1. HA-AP健康状态监控功能测试
      1. ABTS and queue\_full 预警功能测试:进行为期3天的测试，出现过两次意外错误，一个是无法找到VPD文件，但查看历史文件运行正常；另外一Errno 10054是远程机掉链接导致。两次意外错误不会导致程序正常运行，后续测试结果正常
      2. Engine reboot预警功能测试：通过不定期重启引擎，观察并记录程序是否正常运行、有无出现程序报错、运行结果是否匹配。运行结果正常，历史数据写入正常，成功收到预警邮件
   2. Switch 健康状态监控功能测试:故障设备IP获取功能正常、历史数据写入正常、邮件警报发送正常。并且模拟SAN环境中Engine掉线，同组Switch某台掉线对整个环境，功能的影响，目前测试正常
   3. 网页显示界面及用户反馈测试：
      1. 网页界面：SAN数据实时更新正常，导航栏链接也能链接到历史错误文件，SAN亚健康时有网页警报提示
      2. 用户反馈：用户反馈测试正常，能清除当前错误IP，并且停止邮件预警，网页警报清除
2. 已知问题或待优化内容：
   1. 问题
      1. 在GUI每30分钟执行task\_30m时，会通过ssh连接switch，若 连接不能马上建立，GUI面板会有卡顿现象。

原因：GUI的Tkinter面板与ssh连接交换机在一个线程中，若ssh连接不上交换机，则会“挡住”Tkinter的运行。

目前设置了连接的timeout时间，缩短了卡顿时间。后续优化目标：将连接交换机的步骤放在另一个子线程执行。

* + 1. GUI运行预警功能无法检测初次运行GUI时Engine的数据，级少数情况能获取到，导致多次初次运行预警功能无预警反应 。

目前原因尚不确定。

后续优化目标：解决这个bug，初次运行（前30min）时GUI功能一切正常。

* 1. 待优化（功能）：
     1. 现场同一组交换机配置用户名和密码时，所有交换机用户名要相同，密码也要一致；

原因：配置文件中读取交换机用户名和密码的标签都只有一个，用户名都是读username，密码都只读password

[SWSetting]

switchname="brocade" （交换机品牌）

port = 22

username = （交换机用户名）

password = （交换机密码）

后续优化目标：根据现场实际交换机个数，在程序配置文件中增加 username2，password2…等，不用强制统一用户名和密码

* + 1. 本次ABTS&Qfull的判断预警方法是将将所有端口的ABTS和Qfull数据加起来，判断是否大于0。

后续优化目标：

* + - 1. 根据用户需求，看是否将ABTS与Qfull两种数据进行分开判断
      2. 今后若需要调整ABTS & Qfull的预警阀值（在某些端口的ABTS或Qfull值大于某个值时发送警报），可以在配置文件中写入阀值，直接修改即可。
    1. 目前打开和关闭flask server是通过menubar的”Open Flask Server”和“Close Flask Server”完成的。

后续优化目标：通过”Open Flask Server”打开server后，出现GUI弹窗界面（提示flask server已打开），关闭GUI弹窗则关闭Flask server。

* + 1. 网页历史数据的显示是否需要后续优化？

目前是打开链接展示错误历史文件

后续优化目标：能以表格等更易看更友好的格式打开，展示给用户

* + 1. 历史文件中，时间获取的是当前系统的时间，如果系统时间不准确，历史信息时间也不准确

后续优化目标：即使当前系统的时间不准确，也能获取到正确的当前时间

* + 1. 目前预警网页在 IE8浏览器和猎豹浏览器中部分功能不可用，预警弹窗不可用，发生警报时综合信息栏背景颜色无响应（建议用户升级IE浏览器或者更改默认浏览器）

后续优化目标：能支持大部分常见OS中常用的浏览器

* + 1. 原有GUI功能的优化：目前邮件中的location是在配置文件中设定地点

后续优化目标：程序可以自动根据所在地点设定location

* 1. 待优化（代码）：
     1. 优化代码中信号变量SIGNAL, flask\_ptr的结构设计

目前：\_\_builtin\_\_.SIGNAL = [0]; \_\_builtin\_\_.flask\_ptr = [0]

后续优化目标：除了list，有没有其他更适合的data structure？

* + 1. 优化代码，使代码更加精简、明了
    2. 降低代码耦合性，实现模块之间松耦合
    3. 优化代码结构：代码以class形式实现
  1. 经过测试，目前程序功能趋于稳定，暂无其他问题

1. 需安装模块：
   1. paramiko：Python模块，基于SSH用于连接远程服务器并执行相关操作，

可使用pip install paramiko安装（如没pip需先安装pip，可参考Flask框架pip安装.docx）

* 1. Flask框架：网页模块制作，可使用pip install flask安装