使用Python收集IOS-XE路由协议抖动日志

目录

简介

本文档介绍如何配置Python脚本，以便在协议摆动时收集OSPF、EIGRP和IS-IS日志。

先决条件

要求

思科建议您熟悉下列主题：

- 应用托管配置
- OSPF
- EIGRP
- IS-IS
- Vi 编辑器

使用的组件

本文档中的信息基于Cisco IOS XE软件版本17。

本文档中的信息都是基于特定实验室环境中的设备编写的。本文档中使用的所有设备最初均采用原始（默认）配置。如果您的网络处于活动状态，请确保您了解所有命令的潜在影响。

注意：本文档不深入介绍应用详情。有关详细信息，请参阅参考链接。

配置

配置

创建TAC案例时，收集相关信息以节省时间非常重要。有时，故障线索存在于可以从设备收集的一些基本输出中。在本文档中，您有如何利用Python脚本获取此数据的示例。考虑三种协议：OSPF、EIGRP和IS-IS。

步骤1.您需要做的第一件事是配置和启用guestshell。

```
Router(config)#iox
Router(config)#interface VirtualPortGroup 0
Router(config-if)#ip address 192.0.2.1 255.255.255.252
Router(config-if)#exit
Router(config)#
Router(config)#app-hosting appid guestshell
Router(config-app-hosting)#app-vnic gateway0 virtualportgroup 0 guest-interface 0
Router(config-app-hosting-gateway0)#guest-ipaddress 192.0.2.2 netmask 255.255.255.252
Router(config-app-hosting)#app-default-gateway 192.0.2.1 guest-interface 0
Router(config)#end
```

在此配置中，有三个重要步骤：

1.启用IOX服务。启用guestshell需要此命令。

2.配置充当guestshell默认网关的默认网关的VirtualPortGroup。

3.为guestshell配置应用托管。您可以从配置中了解VirtualPortGroup发挥作用的位置。

步骤2.接下来，您需要从特权模式启用guestshell。

```
Router#guestshell enable
Interface will be selected if configured in app-hosting
Please wait for completion
guestshell installed successfully
Current state is: DEPLOYED
guestshell activated successfully
```

```
Current state is: ACTIVATED
guestshell started successfully
Current state is: RUNNING
Guestshell enabled successfully

Router#
*Jun 15 21:31:31.499: %IM-6-IOX_INST_INFO: R0/0: ioxman: IOX SERVICE guestshell LOG: Guestshell is up a
```

如果所有配置都正确，则必须查看上例中的日志。

步骤3.现在，您已准备好配置python脚本。在特权模式下运行guestshell命令。您会看到如下例所示的提示：

```
Router#guestshell
[guestshell@guestshell ~]$
```

步骤4.使用vi编辑器创建文件，并根据已启用的协议配置脚本。

```
[guestshell@guestshell ~]$ vi ospf.py
```

此窗口显示

```
~
~
~
~
~
~
~
"ospf.py" 0L, 0C
```

步骤5.按“i”以插入文本。粘贴脚本，然后按“esc”，然后输入字符:wq

```
~
from cli import cli
from time import sleep

cli("enable")
cli("debug ip ospf hello")
cli("debug ip ospf adj")
cli("show ip ospf interface | append bootflash:Router-ospf-logs.txt")
cli("show ip ospf neighbor | append bootflash:Router-ospf-logs.txt")
cli("show interfaces | append bootflash:Router-ospf-logs.txt")
```

```
cli("show logging | append bootflash:Router-ospf-logs.txt")
cli("show tech | append bootflash:Router-showtech.txt")
sleep(30)
cli("undebug all")
~
~
~
~
"ospf.py" [New] 14L, 458C written
[guestshell@guestshell ~]$
```

使用exit命令退出guestshell。

验证

测试脚本。使用exit命令从guestshell退出。然后运行guestshell run python3 ospf.py

```
F340.20.09-8500-1#guestshell run python3 ospf.py
```

以下是这三个协议的脚本；OSPF、EIGRP和IS-IS。

OSPF

```
from cli import cli
from time import sleep

cli("enable")
cli("debug ip ospf hello")
cli("debug ip ospf adj")
cli("show ip ospf interface | append bootflash:Router-ospf-logs.txt")
cli("show ip ospf neighbor | append bootflash:Router-ospf-logs.txt")
cli("show interfaces | append bootflash:Router-ospf-logs.txt")
cli("show logging | append bootflash:Router-ospf-logs.txt")
cli("show tech | append bootflash:Router-showtech.txt")
sleep(30)
cli("undebug all")
```

EIGRP

```
from cli import cli
from time import sleep

cli("enable")
cli("debug eigrp packet")
```

```
cli("show ip eigrp neighbor | append bootflash:Router-eigrp-logs.txt")
cli("show ip eigrp interface | append bootflash:Router-eigrp-logs.txt")
cli("show interfaces | append bootflash:Router-eigrp-logs.txt")
cli("show logging | append bootflash:Router-eigrp-logs.txt")
cli("show tech | append bootflash:Router-showtech.txt")
sleep(30)
cli("undebug all")
```

IS-IS

```
from cli import cli
from time import sleep

cli("enable")
cli("debug isis adj-packet")
cli("show isis neighbor detail | append bootflash:Router-isis-logs.txt")
cli("show clns neighbor detail | append bootflash:Router-isis-logs.txt")
cli("show clns interface | append bootflash:Router-isis-logs.txt")
cli("show interfaces | append bootflash:Router-isis-logs.txt")
cli("show logging | append bootflash:Router-isis-logs.txt")
cli("show tech | append bootflash:Router-showtech.txt")
sleep(30)
cli("undebug all")
```

您可以在观察系统日志模式后使用运行Python脚本的EEM脚本自动收集日志。在下一节中，您将可以配置EEM脚本以及python脚本来完成此任务。

OSPF

```
event manager applet ospf-flap authorization bypass
event syslog pattern "%OSPF-5-ADJCHG:.*from FULL to DOWN" maxrun 120 ratelimit 120
action 010 cli command "enable"
action 020 cli command "guestshell run python3 ospf.py"
action 030 exit
```

EIGRP

```
event manager applet eirgp-flap authorization bypass
event syslog pattern "%DUAL-5-NBRCHANGE: EIGRP.*Neighbor.*is down" maxrun 120 ratelimit 120
action 010 cli command "enable"
action 020 cli command "guestshell run python3 eigrp.py"
action 030 exit
```

IS-IS

```
event manager applet isis-flap authorization bypass
event syslog pattern "%CLNS-5-ADJCHANGE: ISIS: Adjacency to.*Down" maxrun 120 ratelimit 120
action 010 cli command "enable"
action 020 cli command "guestshell run python3 isis.py"
action 030 exit
```

注意：这些脚本中收集的命令提供基本初始信息。创建TAC支持请求时，TAC工程师可以请求获得更多信息，以便在需要时进行进一步调查。

参考链接

- Guestshell
- Python API

关于此翻译
思科采用人工翻译与机器翻译相结合的方式将此文档翻译成不同语言，希望全球的用户都能通过各自的语言得到支持性的内容。

请注意：即使是最好的机器翻译，其准确度也不及专业翻译人员的水平。

Cisco Systems, Inc. 对于翻译的准确性不承担任何责任，并建议您总是参考英文原始文档（已提供链接）。