了解EEM的最佳实践和有用的脚本

目录

简介

先决条件

- 要求
- 使用的组件

最佳实践

- 确认适当的身份验证已就绪
- 添加EEM运行时和速率限制的约束
- 避免执行顺序混乱
- 禁用分页
- 面向未来可维护性的设计脚本

常见EEM逻辑模式

- 使用If/Else的分支机构代码路径
- 循环语句
- 通过正则表达式(Regex)提取输出

有用的EEM脚本

- 跟踪MAC地址学习的特定MAC地址
- 通过SNMP OID监控高CPU
- 动态匹配PID并记录堆栈输出
- 升级交换机
- 当IP SLA跟踪的对象关闭时，将诊断数据转储到文件
- 从EEM发送电子邮件
- 按计划关闭端口
- 达到给定每秒数据包数(PPS)速率时关闭接口
- 使用CDP邻居自动填充接口描述

相关信息

简介

本文档介绍Cisco IOS® XE设备上的嵌入式事件管理器(EEM)脚本配置最佳实践。

先决条件

要求

思科建议您了解并熟悉以下主题：

- Cisco IOS和Cisco IOS XE嵌入式事件管理器(EEM)

如果您不熟悉此功能，请先阅读EEM功能概述。

使用的组件

本文档中的信息基于以下软件和硬件版本：

- Cisco Catalyst 9300、9400和9500交换机
- Cisco IOS软件版本16.X或17.X

注意：思科TAC不支持这些脚本，这些脚本按原样提供，用于教学目的。

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。

最佳实践

本部分介绍在设计和实施EEM脚本时观察到的一些最常见问题。有关EEM最佳实践的更多信息，请参阅参考部分下面引用的EEM最佳实践文档。

确认适当的身份验证已就绪

如果您的设备使用AAA，您必须确保设备上配置的EEM脚本配置有能够运行脚本中命令的AAA用户，或者授权旁路配置有脚本定义中的authorization bypass命令。

添加EEM运行时和速率限制的约束

默认情况下，EEM脚本最多可运行20秒。如果设计的脚本需要较长时间才能运行，或者必须在命令执行之间等待，请在applet事件触发器上指定maxrun值以更改默认执行计时器。

此外，还必须考虑触发EEM脚本的事件运行的频率。如果触发脚本的条件是在短时间内迅速发生（例如，MAC摆动的系统日志触发器），则在EEM脚本中包含速率限制条件非常重要，这样可防止

并行执行次数过多并防止设备资源耗尽。

避免执行顺序混乱

如EEM文档中所述，action语句的执行顺序由其标签控制(例如，action 0001 cli command enable的标签为0001)。 此标签值不是数字，而是字母数字。操作按升序字母数字键序列进行排序，使用label参数作为排序键，它们按此序列运行。这可能会根据操作标签的结构导致意外的执行顺序。

请考虑以下示例：

```
event manager applet test authorization bypass
event timer watchdog time 60 maxrun 60
action 13 syslog msg "You would expect to see this message first"
action 120 syslog msg "This message prints first"
```

因为在字母数字比较中，120在13之前，所以此脚本不会按您期望的顺序运行。为了避免这种情况，使用如下填充系统很有用：

```
event manager applet test authorization bypass
event timer watchdog time 60 maxrun 60
action 0010 syslog msg "This message appears first"
action 0020 syslog msg "This message appears second"
action 0120 syslog msg "This message appears third"
```

由于此处有填充，编号语句按预期顺序计算。每个标签之间增加10可以允许以后根据需要将其他语句插入到EEM脚本中，而无需对所有后续语句重新编号。

禁用分页

EEM会查找设备提示符以确定命令输出何时完成。输出数据多于一个屏幕可显示的命令（根据终端长度配置）可阻止EEM脚本完成（并最终通过maxrun计时器终止），因为在查看输出的所有页面之前不会显示设备提示。在检查大型输出的EEM脚本开头配置term len 0。

面向未来可维护性的设计脚本

设计EEM脚本时，请留出动作标签之间的间隙，以便将来更轻松地更新EEM脚本逻辑。如果存在适当的间隙(即，诸如action 0010和action 0020之类的语句会留出9个标签的间隙以供插入)，则可以根据需要添加新语句，而无需对操作标签进行重新编号或重新检查，并确保操作继续按预期顺序执行

。

您需要在EEM脚本开始时运行一些常用命令。这可能包括：

- 将终端长度设置为0
- 进入启用模式
- 启用命令输出的自动时间戳

这是本文档所示示例中的常见模式，其中许多脚本都以相同的3个action语句开始配置此脚本。

常见EEM逻辑模式

本节介绍EEM脚本中使用的一些常见逻辑模式和语法块。此处的示例不是完整的脚本，而是说明如何使用特定功能创建复杂的EEM脚本的演示。

使用If/Else的分支机构代码路径

EEM变量可用于控制EEM脚本的执行流。考虑以下EEM脚本：

```
event manager applet snmp_cpu authorization bypass
event timer watchdog time 60
action 0010 info type snmp oid 1.3.6.1.4.1.9.9.109.1.1.1.1.3 get-type exact
action 0020 if $_info_snmp_value ge "50"
action 0030 syslog msg "This syslog message is sent if CPU utilization is above 50%"
action 0040 elseif $_info_snmp_value ge "30"
action 0050 syslog msg "This syslog message is sent if CPU utilization is above 30% and below 50%"
action 0060 else
action 0070 syslog msg "This syslog message is sent if CPU utilization is below 30%"
action 0080 end
```

此脚本每分钟运行一次。检查SNMP OID的CPU使用率值，然后根据OID的值输入三个不同执行路径之一。类似的语句可用于任何其他合法EEM变量，以在EEM脚本中构建复杂的执行流。

循环语句

执行循环可用来显着缩短EEM脚本，并使其更易于推理。请考虑以下脚本，该脚本旨在将Te2/1/15的接口统计信息在1分钟内拉动6次，以检查是否存在小时间段的高利用率：

```
event manager applet int_util_check auth bypass
```

```
event timer watchdog time 300 maxrun 120
action 0001 cli command "enable"
action 0002 cli command "term exec prompt timestamp"
action 0003 cli command "term length 0"
action 0010 syslog msg "Running iteration 1 of command"
action 0020 cli command "show interface te2/1/15 | append flash:interface_util.txt"
action 0030 wait 10
action 0040 syslog msg "Running iteration 2 of command"
action 0050 cli command "show interface te2/1/15 | append flash:interface_util.txt"
action 0060 wait 10
action 0070 syslog msg "Running iteration 3 of command"
action 0080 cli command "show interface te2/1/15 | append flash:interface_util.txt"
action 0090 wait 10
action 0100 syslog msg "Running iteration 4 of command"
action 0110 cli command "show interface te2/1/15 | append flash:interface_util.txt"
action 0120 wait 10
action 0130 syslog msg "Running iteration 5 of command"
action 0140 cli command "show interface te2/1/15 | append flash:interface_util.txt"
action 0150 wait 10
action 0160 syslog msg "Running iteration 6 of command"
action 0170 cli command "show interface te2/1/15 | append flash:interface_util.txt"
```

使用EEM循环结构时，可以显着缩短此脚本：

```
event manager applet int_util_check auth bypass
event timer watchdog time 300 maxrun 120
action 0001 cli command "enable"
action 0002 cli command "term exec prompt timestamp"
action 0003 cli command "term length 0"
action 0010 set loop_iteration 1
action 0020 while $loop_iteration le 6
action 0030 syslog msg "Running iteration $loop_iteration of command"
action 0040 cli command "show interface te2/1/15 | append flash:interface_util.txt"
action 0050 wait 10
action 0060 increment loop_iteration 1
action 0070 end
```

通过正则表达式(Regex)提取输出

EEM regexp语句可用于从命令输出中提取值，用于后续命令并在EEM脚本本身中启用动态命令创建。有关从show proc cpu输出提取SNMP引擎PID的示例，请参阅此代码块 | i SNMP引擎并将其打印到系统日志消息。此提取的值也可用于需要PID运行的其他命令中。

```
event manager applet check_pid auth bypass
event none
action 0010 cli command "show proc cpu | i SNMP ENGINE"
action 0020 regexp "^[ ]*([0-9]+) .*" $_cli_result match match1
action 0030 syslog msg "Found SNMP Engine PID $match1"
```

有用的EEM脚本

跟踪MAC地址学习的特定MAC地址

在本示例中，跟踪MAC地址b4e9.b0d3.6a41。脚本每30秒检查一次，以查看是否已在ARP或MAC表中获取了指定的MAC地址。如果看到MAC，脚本将执行以下操作：

- 输出系统日志消息（当您要确认获取MAC地址的位置或获取时间/频率时，这很有用）。

实现

```
event manager applet mac_trace authorization bypass
event timer watchdog time 30
action 0001 cli command "enable"
action 0002 cli command "term exec prompt timestamp"
action 0003 cli command "term length 0"
action 0010 cli command "show ip arp | in b4e9.b0d3.6a41"
action 0020 regexp ".*(ARPA).*" $_cli_result
action 0030 if $_regexp_result eq 1
action 0040 syslog msg $_cli_result
action 0050 end
action 0060 cli command "show mac add vlan 1 | in b4e9.b0d3.6a41"
action 0070 regexp ".*(DYNAMIC).*" $_cli_result
action 0080 if $_regexp_result eq 1
action 0090 syslog msg $_cli_result
action 0100 end
```

通过SNMP OID监控高CPU

此脚本监控用于读取最后5秒内CPU忙碌百分比的SNMP OID。当CPU繁忙超过80%时，脚本将采取以下操作：

- 根据show clock的输出创建时间戳，并使用此创建唯一的文件名。
- 然后将关于进程和软件状态的输出写入此文件。
- 嵌入式数据包捕获(EPC)配置为捕获发往控制平面的10秒流量并将其写入文件。
- 完成EPC捕获后，EPC配置将被删除，脚本将退出。

实现

```
event manager applet high-cpu authorization bypass
event snmp oid 1.3.6.1.4.1.9.9.109.1.1.1.1.3 get-type next entry-op gt entry-val 80 poll-interval 1 rat
```

```
action 0001 cli command "enable"
action 0002 cli command "term exec prompt timestamp"
action 0003 cli command "term length 0"
action 0010 syslog msg "High CPU detected, gathering system information."
action 0020 cli command "show clock"
action 0030 regex "([0-9]|[0-9][0-9]):([0-9]|[0-9][0-9]):([0-9]|[0-9][0-9])" $_cli_result match match1
action 0040 string replace "$match" 2 2 "."
action 0050 string replace "$_string_result" 5 5 "."
action 0060 set time $_string_result
action 0070 cli command "show proc cpu sort | append flash:tac-cpu-$time.txt"
action 0080 cli command "show proc cpu hist | append flash:tac-cpu-$time.txt"
action 0090 cli command "show proc cpu platform sorted | append flash:tac-cpu-$time.txt"
action 0100 cli command "show interface | append flash:tac-cpu-$time.txt"
action 0110 cli command "show interface stats | append flash:tac-cpu-$time.txt"
action 0120 cli command "show log | append flash:tac-cpu-$time.txt"
action 0130 cli command "show ip traffic | append flash:tac-cpu-$time.txt"
action 0140 cli command "show users | append flash:tac-cpu-$time.txt"
action 0150 cli command "show platform software fed switch active punt cause summary | append flash:tac
action 0160 cli command "show platform software fed switch active cpu-interface | append flash:tac-cpu-
action 0170 cli command "show platform software fed switch active punt cpuq all | append flash:tac-cpu-
action 0180 cli command "no monitor capture tac_cpu"
action 0190 cli command "monitor capture tac_cpu control-plane in match any file location flash:tac-cpu
action 0200 cli command "monitor capture tac_cpu start" pattern "yes"
action 0210 cli command "yes"
action 0220 wait 10
action 0230 cli command "monitor capture tac_cpu stop"
action 0240 cli command "no monitor capture tac_cpu"
```

动态匹配PID并记录堆栈输出

此脚本查找SNMP输入队列已满的系统日志消息并采取以下操作：

- 将show proc cpu sort的输出记录到文件中。
- 通过regex提取SNMP引擎进程的PID。
- 在后续命令中使用SNMP PID来获取PID的堆栈数据。
- 从配置中删除脚本，以便不再执行该脚本。

实现

```
event manager applet TAC-SNMP-INPUT-QUEUE-FULL authorization bypass
event syslog pattern "INPUT_QFULL_ERR" ratelimit 40 maxrun 120
action 0010 cli command "en"
action 0020 cli command "show proc cpu sort | append flash:TAC-SNMP.txt"
action 0030 cli command "show proc cpu | i SNMP ENGINE"
action 0040 regexp "^[ ]*([0-9]+) .*" $_cli_result match match1
action 0050 syslog msg "Found SNMP Engine PID $match1"
action 0060 cli command "show stacks $match1 | append flash:TAC-SNMP.txt"
action 0070 syslog msg "$_cli_result"
action 0080 cli command "configure terminal"
action 0090 cli command "no event manager applet TAC-SNMP-INPUT-QUEUE-FULL"
action 0100 cli command "end"
```

升级交换机

此脚本配置为在install add file <file> activate commit命令返回的非标准提示符上执行模式匹配并响应提示。未配置触发事件，因此，当需要通过运行UPGRADE的事件管理器进行升级时，必须由用户手动触发EEM脚本。maxrun计时器设置为300秒，而不是默认值20秒，因为install add命令需要大量时间运行。

实现

```
event manager applet UPGRADE authorization bypass
event none maxrun 300
action 0001 cli command "enable"
action 0002 cli command "term length 0"
action 0020 cli command "install add file flash:cat9k_iosxe.16.06.02.SPA.bin activate commit" pattern "
action 0030 cli command "y" pattern "y\/n"
action 0040 syslog msg  "Reloading device to upgrade code"
action 0050 cli command "y"
```

当IP SLA跟踪的对象关闭时，将诊断数据转储到文件

当IP SLA对象11关闭并执行下列操作时，会触发此脚本：

- 收集MAC表、ARP表、系统日志和路由表。
- 将信息写入闪存中的文件：称为sla_track.txt。

实现

```
ip sla 10
icmp-echo 10.10.10.10 source-ip 10.10.10.10
frequency 10
exit
ip sla schedule 10 life forever start-time now
track 11 ip sla 10 reachability
exit
event manager applet track-10 authorization bypass
event track 11 state down
action 0001 cli command "enable"
action 0002 cli command "term exec prompt timestamp"
action 0003 cli command "term length 0"
action 0010 syslog msg  "IP SLA object 10 has gone down"
action 0020 cli command "show mac address-table detail | append flash:sla_track.txt"
action 0030 cli command "show ip arp | append flash:sla_track.txt"
```

```
action 0040 cli command "show log | append flash:sla_track.txt"
action 0050 cli command "show ip route | append flash:sla_track.txt"
```

从EEM发送电子邮件

此脚本在看到event syslog pattern语句中描述的模式时触发，并执行以下操作：

- 从内部电子邮件服务器发送电子邮件（假设内部电子邮件服务器允许从设备进行开放式身份验证）。

实现

```
event manager environment email_from email_address@company.test
event manager environment email_server 192.168.1.1
event manager environment email_to dest_address@company.test
event manager applet email_syslog
event syslog pattern "SYSLOG PATTERN HERE" maxrun 60
action 0010 info type routername
action 0020 mail server "$email_server" to "$email_to" from "$email_from" subject "SUBJECT OF EMAIL - S
```

按计划关闭端口

此脚本在每天下午6点关闭端口Te2/1/15。

实现

```
event manager applet shut_port authorization bypass
event timer cron cron-entry "0 18 * * *"
action 0001 cli command "enable"
action 0002 cli command "term exec prompt timestamp"
action 0003 cli command "term length 0"
action 0010 syslog msg  "shutting port Te2/1/15 down"
action 0030 cli command "config t"
action 0040 cli command "int Te2/1/15"
action 0050 cli command "shutdown"
action 0060 cli command "end"
```

达到给定每秒数据包数(PPS)速率时关闭接口

此脚本每秒在TX方向上检查接口Te2/1/9上的PPS速率。如果PPS速率超过100，它会采取以下措施：

- 将接口的输出记录到syslogshow int。
- 关闭接口。

实现

```
event manager applet disable_link authorization bypass
event interface name te2/1/9  parameter  transmit_rate_pps entry-op ge entry-val 100 poll-interval 1 en
action 0001 cli command "enable"
action 0002 cli command "term length 0"
action 0010 syslog msg  "Detecting high input rate on interface te2/1/9.  Shutting interface down."
action 0020 cli command "show int te2/1/9"
action 0030 syslog msg  $_cli_result
action 0040 cli command "config t"
action 0050 cli command "int te2/1/9"
action 0060 cli command "shutdown"
action 0070 cli command "end"
```

使用CDP邻居自动填充接口描述

此脚本定期检查CDP邻居信息，并根据连接的CDP邻居设备自动更新本地接口描述。该脚本每5分钟运行一次，并执行以下操作：

- 执行 show cdp neighbors detail.
- 解析设备ID和本地接口。
- 进入每个已发现接口的接口配置模式。
- 将接口描述设置为Connected to <CDP Neighbor>。
- 生成确认更新的系统日志消息。

这对于在链路经常变化或需要一致文档的环境中保持准确的接口描述非常有用。

实现

```
event manager applet CDP-Neighbor-Description
event timer watchdog time 300
event none
action 1.0 cli command "enable"
```

```
action 2.0 cli command "show cdp neighbors detail"
action 3.0 set temp_device ""
action 4.0 foreach line "$_cli_result" "\n"
action 4.1 regexp "Device ID: ([^\r\n]+)" "$line" match device_id
action 4.2 if $_regexp_result eq "1"
action 4.3 set temp_device "$device_id"
action 4.4 end
action 4.5 regexp "Interface: ([^,]+)," "$line" match intf
action 4.6 if $_regexp_result eq "1"
action 4.7 if "$temp_device" ne ""
action 4.71 cli command "configure terminal"
action 4.72 cli command "interface $intf"
action 4.73 cli command "description Connected to $temp_device"
action 4.74 cli command "end"
action 4.75 syslog msg "EEM: Set description on $intf to 'Connected to $temp_device'"
action 4.76 set temp_device ""
action 4.8 end
action 4.9 end
action 5.0 end
```

相关信息

- 思科EEM最佳实践
- 思科技术支持和下载

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）。