自动响应系统事件

本章介绍如何配置嵌入式事件管理器 (EEM)。

关于 EEM

EEM 服务使您可以调试问题并提供用于故障排除的通用日志记录。这项服务由两个部分组成：EEM 响应或侦听的事件，以及定义操作和 EEM 所响应事件的事件管理器小程序。您可以配置多个事件管理器小程序来响应不同的事件和执行不同的操作。

支持的事件

EEM 支持以下事件：

- 系统日志 - ASA 使用系统日志消息 ID 标识触发事件管理器小程序的系统日志消息。您可以配置多个系统日志事件，但系统日志消息 ID 可能不会在一个事件管理器小程序内重叠。
- 计时器 - 可以使用计时器触发事件。对于每个事件管理器小程序，每个计时器只能配置一次。每个事件管理器小程序最多可以有三个计时器。计时器的三种类型如下：
  - 看门狗（定期）计时器在小程序操作完成后的指定时间段后触发事件管理器小程序，并会自动重新启动。
  - 倒数（一次性）计时器在指定时间段后立即触发事件管理器小程序，且通常不会重新启动，除非删除并重新添加它们。
  - 绝对（一天一次）计时器促使事件在每天的指定时间发生一次，并会自动重新启动。时间格式为 hh:mm:ss。

对于上述类型的每个事件管理器小程序，只能配置一个计时器事件。

- 无 - 当您使用 CLI 或 ASDM 手动运行事件管理器小程序时，会触发 None 事件。
- 故障 - 当 ASA 出现故障时，触发故障事件。不管 **output** 命令的值是什么， **action** 命令都会定向至 crashinfo 文件。输出会在 **show tech** 命令之前生成。

事件管理器小程序上的操作

当事件管理器小程序被触发时，会执行事件管理器小程序上的操作。每个操作都具有用于指定操作序列的编号。该序列号在事件管理器小程序中必须是唯一的。您可以为一个事件管理器小程序配置多个操作。命令是典型的 CLI 命令，例如 **show blocks**。

输出目标

您可以使用 **output** 命令将操作输出发送到指定的位置。一次只能启用一个输出值。默认值为 **output none**。此值会丢弃 **action** 命令的任何输出。命令在全局配置模式下作为权限级别为 15（最高）的用户来运行。此命令可能不接受任何输入，因为它处于禁用状态。您可以将 **action** CLI 命令的输出发送到以下三个位置之一：

- **None** - 这是默认位置，会丢弃输出
- **Console** - 此位置将输出发送到 ASA 控制台
- **File** - 此位置将输出发送到文件。以下四个文件选项可用：
  - **Create a unique file** - 每次调用事件管理器小程序时，此选项会创建具有唯一名称的新文件
  - **Create/overwrite a file** - 每次调用事件管理器小程序时，此选项会覆盖指定的文件。
  - **Create/append to a file** - 每次调用事件管理器小程序时，此选项会附加到指定的文件。如果指定的文件不存在，则会创建文件。
  - **Create a set of files** - 此选项会创建一组具有唯一名称的文件，每次调用事件管理器小程序时，都会轮换这些文件。

EEM 准则

本节介绍在配置 EEM 之前应检查的准则和限制。

情景模式准则

不支持多情景模式。

其他规定

- 在发生崩溃期间，ASA 的状态一般是未知的。在这种情况下运行某些命令可能不安全。

- 事件管理器小程序的名称不能包含空格。
- 不能修改 None 事件和 Crashinfo 事件参数。
- 因为系统日志消息会发送到 EEM 中进行处理，因此可能会影响性能。
- 每个事件管理器小程序的默认输出均为 **output none**。要更改此设置，必须输入其他输出值。
- 只能为每个事件管理器小程序定义一个输出选项。

配置 EEM

EEM 的配置由以下任务组成：

过程

步骤 1 创建事件管理器小应用程序并配置事件，第 3 页。
步骤 2 配置操作和操作输出的目标，第 5 页。
步骤 3 运行事件管理器小程序，第 7 页。
步骤 4 跟踪内存分配和内存使用，第 7 页。

创建事件管理器小应用程序并配置事件

要创建事件管理器小程序并配置事件，请执行以下步骤：

过程

步骤 1 创建事件管理器小程序并进入事件管理器小程序配置模式。

event manager applet 名称

示例：

```
ciscoasa(config)# event manager applet exampleapplet1
```

name 参数最多可包含 32 个字母数字字符。不允许使用空格。

要删除事件管理器小应用程序，请输入此命令的 **no** 形式。

步骤 2 描述事件管理器小程序。

description 文本

示例：

```
ciscoasa(config-applet)# description applet1example
```

text 参数最多可以包含 256 个字符如果用引号将说明文本引起来，说明文本可包含空格。

步骤 3 要配置指定事件，请输入以下命令之一。要删除已配置的事件，请输入相应命令的 **no** 形式。

- 要配置系统日志事件，请确定一条或一系列触发事件管理器小程序的系统日志消息。

  event syslog id *nnnnnn* [*-nnnnnn*] [**occurs** *n*] [**period** *seconds*]

  示例：

  ```
  ciscoasa(config-applet)# event syslog id 106201
  ```

  nnnnnn 参数标识系统日志消息 ID。**occurs** *n* 关键字-参数对指示系统日志消息必须发生多长时间，事件管理器小应用程序 才会被调用。默认情况为每 0 秒出现 1 次。有效值为 1 到 4294967295。**period** *seconds* 关键字-参数对指示事件必须发生的秒数， 并将事件管理器小应用程序 的调用频率限制为在配置的时间内最多调用一次。有效值为 0 到 604800。值 0 表示未定义时间段。

- 要将事件配置为在每个配置的时间段内发生一次并自动重新启动，请输入以下命令。

  event timer watchdog time 秒

  示例：

  ```
  ciscoasa(config-applet)# event timer watchdog time 30
  ```

  秒数的范围为 1 - 604800。

- 要将事件配置为发生一次且不会重新启动（除非删除然后重新添加事件），请输入以下命令。

  event timer countdown time 秒

  示例：

  ```
  ciscoasa(config-applet)# event timer countdown time 60
  ```

  秒数的范围为 1 - 604800。使用此命令的 **no** 形式会删除倒计时计时器事件。

  注释 如果这是启动配置，当您重新启动时此计时器将会重新运行。

- 要将事件配置为在指定时间一天发生一次并自动重新启动，请输入以下命令。

  event timer absolute time 小时:分钟:秒

  示例：

  ```
  ciscoasa(config-applet)# event timer absolute time 10:30:20
  ```

  时间格式为 hh:mm:ss。事件范围为 00:00:00（午夜）至 23:59:59。

- ASA 出现崩溃时会触发崩溃事件。

  event crashinfo

  示例：

  ```
  ciscoasa(config-applet)# event crashinfo
  ```

  不管 **output** 命令的值是什么， **action** 命令都会定向至 crashinfo 文件。输出会在 **show tech** 命令之前生成。

配置操作和操作输出的目标

要配置操作和操作输出的特定发送目标，请执行以下步骤：

过程

步骤 1 在事件管理器小程序上配置操作。

action *n* **cli command** “*command*”

示例：

```
ciscoasa(config-applet)# action 1 cli command “show version”
```

n 选项是操作 ID。有效 ID 的范围为 0 到 4294967295。*command* 选项的值必须位于引号中；否则，如果命令由多个单词组成，则会发生错误。命令在全局配置模式下作为权限级别为 15（最高）的用户来运行。此命令可能不接受任何输入，因为它处于禁用状态。如果命令可用，请使用选项。**noconfirm**

步骤 2 选择一个可用的输出目标选项。使用相应命令的 **no** 形式可删除输出目标。

- **None** 选项会丢弃 **action** 命令的任何输出（这是默认设置）：

  output none

  示例：

  ```
  ciscoasa(config-applet)# output none
  ```

- **Console** 选项将 **action** 命令的输出发送到控制台。

  output console

  示例：

  ```
  ciscoasa(config-applet)# output console
  ```

注释　运行此命令会影响性能。

- **New File** 选项为调用的每个事件管理器小程序将 **action** 命令的输出发送到新文件。

  output file new

  示例：

  ```
  ciscoasa(config-applet)# output file new
  ```

  文件名的格式为 eem-*applet-timestamp*.log，其中， *applet* 是事件管理器小程序的名称， *timestamp* 是注有日期的时间戳，其格式为 YYYYMMDD-hhmmss。

- **New Set of Rotated Files** 选项可创建一组会轮换的文件。当要写入新文件时，最旧的文件会被删除，且所有的后续文件都会在写入第一个文件之前进行重新编号。

  output file rotate *n*

  示例：

  ```
  ciscoasa(config-applet)# output file rotate 50
  ```

  最新的文件以 0 表示，最旧的文件 以最高编号 (*n*-1) 表示。*n* 选项是轮换值。有效值范围为 2 到 100。文件名格式为 eem-*applet-x*.log，其中，*applet* 是小程序的名称，*x* 是文件编号。

- **Single Overwritten File** 选项将 **action** 命令输出写入到一个文件中，每次写入都会覆盖原有文件。

  output file overwrite *filename*

  示例：

  ```
  ciscoasa(config-applet)# output file overwrite examplefile1
  ```

  filename 参数是本地（至 ASA）文件名。此命令也可以使用 FTP、TFTP 和 SMB 目标文件。

- **Single Appended File** 选项将 **action** 命令输出写入到一个文件中，每次写入时都会附加到原有文件。

  output file append *filename*

  示例：

  ```
  ciscoasa(config-applet)# output file append examplefile1
  ```

  filename 参数是本地（对于 ASA 而言）文件名。

运行事件管理器小程序

要运行事件管理器小程序，请执行以下步骤：

过程

运行事件管理器小程序。

event manager run *applet*

示例：

```
ciscoasa# event manager run exampleapplet1
```

如果运行尚未配置 **event none** 命令的事件管理器小程序，将会发生错误。*applet* 参数是事件管理器小程序的名称。

跟踪内存分配和内存使用

要记录内存分配和内存使用情况，请执行以下步骤：

过程

步骤 1 启用内存日志记录。

memory logging [1024-4194304] [**wrap**] [**size** [1-2147483647]] [**process** *process-name*] [**context** *context-name*]

示例：

```
ciscoasa(config)# memory logging 202980
```

唯一必填参数是内存日志记录缓冲区中的条目数。**wrap** 选项用于通知内存日志记录实用程序在封装时保存缓冲区。缓冲区只能保存一次。

如果内存日志记录缓冲区 wrap 多次，会被覆写。当缓冲区 wrap 时，系统会将触发器发送到事件管理器，以启用数据保存。**size** 选项用于监控特定大小。**process** 选项用于监控特定流程。

注释 Checkheaps 进程被当作一个进程完全忽略，因为它以非标准方式使用内存分配器。

context 选项按给定名称为给定虚拟情景记录内存日志记录。

要更改内存日志记录参数，必须将其禁用，然后重新启用。

步骤 2 显示内存日志记录结果。

show memory logging [**brief** | **wrap**]
show memory logging include [**address**] [**caller**] [**operator**] [**size**] [**process**] [**time**] [**context**]

示例：

```
ciscoasa# show memory logging
Number of free                          6
Number of calloc                        0
Number of malloc                        8
Number of realloc-new                   0
Number of realloc-free                  0
Number of realloc-null                  0
Number of realloc-same                  0
Number of calloc-fail                   0
Number of malloc-fail                   0
Number of realloc-fail                  0
Total operations 14
Buffer size: 50 (3688 x2 bytes)
process=[ci/console] time=[13:26:33.407] oper=[malloc]
addr=0x00007fff2cd0a6c0 size=72 @ 0x00000000016466ea 0x0000000002124542
0x000000000131911a 0x0000000000442bfd process=[ci/console] time=[13:26:33.407] oper=[free]
addr=0x00007fff2cd0a6c0 size=72 @ 0x00000000021246ef 0x00000000013193e8
0x0000000000443455 0x0000000001318f5b
process=[CMGR Server Process] time=[13:26:35.964] oper=[malloc]
addr=0x00007fff2cd0aa00 size=16 @ 0x00000000016466ea 0x0000000002124542
0x000000000182774d 0x000000000182cc8a process=[CMGR Server Process]
time=[13:26:35.964] oper=[malloc]
addr=0x00007fff224bb9f0 size=512 @ 0x00000000016466ea 0x0000000002124542
0x0000000000bfef9a 0x0000000000bff606 process=[CMGR Server Process]
time=[13:26:35.964] oper=[free]
addr=0x00007fff224bb9f0 size=512 @ 0x00000000021246ef 0x0000000000bff3d8
0x0000000000bff606 0x000000000182ccb0
process=[CMGR Server Process] time=[13:26:35.964] oper=[malloc]
addr=0x00007fff224b9460 size=40 @ 0x00000000016466ea 0x0000000002124542
0x0000000001834188 0x000000000182ce83
process=[CMGR Server Process] time=[13:26:37.964] oper=[free]
addr=0x00007fff2cd0aa00 size=16 @ 0x00000000021246ef 0x0000000001827098
0x000000000182c08d 0x000000000182c262 process=[CMGR Server Process]
time=[13:26:37.964] oper=[free]
addr=0x00007fff224b9460 size=40 @ 0x00000000021246ef 0x000000000182711b
0x000000000182c08d 0x000000000182c262 process=[CMGR Server Process]
time=[13:26:38.464] oper=[malloc]
addr=0x00007fff2cd0aa00 size=16 @ 0x00000000016466ea 0x0000000002124542
0x000000000182774d 0x000000000182cc8a process=[CMGR Server Process]
time=[13:26:38.464] oper=[malloc]
addr=0x00007fff224bb9f0 size=512 @ 0x00000000016466ea 0x0000000002124542
0x0000000000bfef9a 0x0000000000bff606 process=[CMGR Server Process]
time=[13:26:38.464] oper=[free]
addr=0x00007fff224bb9f0 size=512 @ 0x00000000021246ef 0x0000000000bff3d8
0x0000000000bff606 0x000000000182ccb0
process=[CMGR Server Process] time=[13:26:38.464] oper=[malloc]
addr=0x00007fff224b9460 size=40 @ 0x00000000016466ea 0x0000000002124542
0x0000000001834188 0x000000000182ce83
process=[ci/console] time=[13:26:38.557] oper=[malloc]
addr=0x00007fff2cd0a6c0 size=72 @ 0x00000000016466ea 0x0000000002124542
0x000000000131911a 0x0000000000442bfd process=[ci/console] time=[13:26:38.557] oper=[free]
addr=0x00007fff2cd0a6c0 size=72 @ 0x00000000021246ef 0x00000000013193e8
0x0000000000443455 0x0000000001318f5b

ciscoasa# show memory logging include process operation size
Number of free                          6
```

```
Number of calloc                          0
Number of malloc                          8
Number of realloc-new                     0
Number of realloc-free                    0
Number of realloc-null                    0
Number of realloc-same                    0
Number of calloc-fail                     0
Number of malloc-fail                     0
Number of realloc-fail                    0
Total operations 14
Buffer size: 50 (3688 x2 bytes)
process=[ci/console] oper=[malloc] size=72 process=[ci/console] oper=[free]
size=72 process=[CMGR Server Process] oper=[malloc] size=16
process=[CMGR Server Process] oper=[malloc] size=512 process=[CMGR Server Process]
oper=[free] size=512 process=[CMGR Server Process] oper=[malloc] size=40
process=[CMGR Server Process] oper=[free] size=16 process=[CMGR Server Process]
oper=[free] size=40 process=[CMGR Server Process] oper=[malloc] size=16
process=[CMGR Server Process] oper=[malloc] size=512 process=[CMGR Server Process]
oper=[free] size=512 process=[CMGR Server Process] oper=[malloc] size=40
process=[ci/console] oper=[malloc] size=72 process=[ci/console]
oper=[free] size=72 ciscoasa# show memory logging brief
Number of free                            6
Number of calloc                          0
Number of malloc                          8
Number of realloc-new                     0
Number of realloc-free                    0
Number of realloc-null                    0
Number of realloc-same                    0
Number of calloc-fail                     0
Number of malloc-fail                     0
Number of realloc-fail                    0
Total operations 14
Buffer size: 50 (3688 x2 bytes)
```

无需任何选项，**show memory logging** 即可显示统计信息以及记录的操作。**brief** 选项仅显示统计信息。**wrap** 选项显示封装后的缓冲区，然后清除数据，以免出现重复数据或保存重复数据。**include** 选项仅包含输出中的指定字段。您可以按任意顺序指定字段，但它们始终以下列顺序显示：

1. Process
2. Time
3. 情景（除非在单模式下）
4. 操作（free/malloc/等）
5. Address
6. Size
7. Callers

输出格式如下：

process=[XXX] time=[XXX] context=[XXX] oper=[XXX] address=0xXXXXXXXX size=XX @ XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX

最多显示 4 个主叫方地址。操作类型列于示例所示的输出（...的数量）中。

步骤 3 响应内存日志记录包装事件。

event memory-logging-wrap

示例：

```
ciscoasa(config)# event manager applet memlog
ciscoasa(config)# event memory-logging-wrap
ciscoasa(config)# action 0 cli command "show memory logging wrap"
ciscoasa(config)# output file append disk0:/memlog.log
```

示例显示了记录所有内存分配的小程序。当为内存日志记录启用包装时，内存日志记录器向事件管理器发送事件以触发配置的小程序。

EEM 示例

以下示例显示这样的事件管理器小程序：每小时记录一次有关阻止泄露情况信息，并将输出写入到一组会轮换的日志文件中，从而保存一天的日志：

```
ciscoasa(config)# event manager applet blockcheck
ciscoasa(config-applet)# description “Log block usage”
ciscoasa(config-applet)# event timer watchdog time 3600
ciscoasa(config-applet)# output rotate 24
ciscoasa(config-applet)# action 1 cli command “show blocks old”
```

以下示例显示这样的事件管理器小程序：在每天凌晨 1 点重新启动 ASA，根据需要保存配置：

```
ciscoasa(config)# event manager applet dailyreboot
ciscoasa(config-applet)# description “Reboot every night”
ciscoasa(config-applet)# event timer absolute time 1:00:00
ciscoasa(config-applet)# output none
ciscoasa(config-applet)# action 1 cli command “reload save-config noconfirm”
```

以下示例显示在午夜与凌晨 3 点之间禁用给定接口的事件管理器小程序。

```
ciscoasa(config)# event manager applet disableintf
ciscoasa(config-applet)# description “Disable the interface at midnight”
ciscoasa(config-applet)# event timer absolute time 0:00:00
ciscoasa(config-applet)# output none
ciscoasa(config-applet)# action 1 cli command “interface GigabitEthernet 0/0”
ciscoasa(config-applet)# action 2 cli command “shutdown”
ciscoasa(config-applet)# action 3 cli command “write memory”

ciscoasa(config)# event manager applet enableintf
ciscoasa(config-applet)# description “Enable the interface at 3am”
ciscoasa(config-applet)# event timer absolute time 3:00:00
ciscoasa(config-applet)# output none
ciscoasa(config-applet)# action 1 cli command “interface GigabitEthernet 0/0”
ciscoasa(config-applet)# action 2 cli command “no shutdown”
```

```
ciscoasa(config-applet)# action 3 cli command “write memory”
```

监控 EEM

请参阅以下命令以监控 EEM：

- **clear configure event manager**

  此命令可删除事件管理器的运行配置。

- **clear configure event manager applet** *appletname*

  此命令可从配置中删除已命名的事件管理器小程序。

- **show counters protocol eem**

  此命令可显示事件管理器的计数器。

- **show event manager**

  此命令可显示有关已配置的事件管理器小程序的信息，包括命中次数和上一次调用事件管理器小程序的时间。

- **show memory logging**、**show memory logging include**

  这些命令可显示关于内存分配和内存使用情况的统计信息。

- **show running-config event manager**

  此命令可显示事件管理器的运行配置。

EEM 历史记录

表 *1: EEM* 历史记录

功能名称	平台版本	说明
嵌入式事件管理器 (EEM)	9.2(1)	EEM服务使您可以调试问题并提供用于故障排除的通用日志记录。这项服务由两个部分组成：EEM 响应或侦听的事件，以及定义操作和 EEM 所响应事件的事件管理器小程序。您可以配置多个事件管理器小程序来响应不同的事件和执行不同的操作。 引入或修改了以下命令：**event manager applet**、**description**、**event syslog id**、**event none**、**event timer** {**watchdog time** *seconds* | **countdown time** *seconds* | **absolute time** *hh:mm:ss*}、**event crashinfo**、**action cli command**、**output** {**none** | **console** | **file** {**append** *filename* | **new** | **overwrite** *filename* | **rotate** *n*}}、**show running-config event manager**、**event manager run**、**show event manager**、**show counters protocol eem**、**clear configure event manager**、**debug event manager**、**debug menu eem**。
EEM 的内存跟踪	9.4(1)	添加了一项新的调试功能来记录内存分配和内存使用情况，以响应内存日志记录封装事件。 我们引入或修改了以下命令：**memory logging**、**show memory logging**、**show memory logging include** 和 **event memory-logging-wrap**。