ASA嵌入式事件管理器配置示例

目錄

簡介

本文檔介紹嵌入式事件管理器(EEM)，它是在Adaptive Security Appliance(ASA)版本9.2(1)中新增的故障排除工具。 功能與Cisco IOS類似?基於EEM。它是基於ASA事件(syslogs)運行CLI命令並儲存輸出的有力方法。本文檔介紹該功能以及一些示例EEM小程式。

必要條件

需求

使用EEM需要將ASA配置為單情景模式。

採用元件

本文檔中的資訊基於ASA 9.2(1)版或更高版本。

准則和限制

本節包含此功能的准則和限制。

上下文模式准則

EEM當前僅在單情景模式下運行的ASA防火牆上受支援。當前不支援在多情景模式下配置的防火牆。

防火牆模式指南

路由和透明防火牆模式目前均支援EEM。

其他准則

- 當裝置崩潰時，ASA的狀態通常未知。當ASA處於此狀態時，某些命令可能無法安全運行。
- 事件管理器applet的名稱不能包含空格。
- 不能修改None事件和Crashinfo事件引數。
- 效能可能會受到影響，因為系統日誌消息被傳送到EEM以進行處理。
- 每個事件管理器小程式的預設輸出是**output none**。要更改預設輸出，必須輸入不同的輸出值。
- 您只能為每個事件管理器小程式定義一個輸出選項。

設定

event manager applet命令建立/編輯事件管理器小程式，該程式將事件與操作和輸出相關聯。*<name>*限製為32個字元，不能包含空格。這將進入事件管理器小程式子模式。

```
ASA(config)# [no] event manager applet
```

可以將**description**新增到applet中。這只是為了提供資訊。*<text>限*製為256個字元。

```
ASA(config-applet)# [no] description
```

事件配置

可將各種事件新增到小程式，這些事件觸發小程式呼叫在其上配置的操作。它們使用**event**關鍵字定義。可以為每個applet配置多個事件。

系統日誌事件

支援的第一個事件型別是**syslog**。ASA使用系統日誌ID來識別觸發小程式的系統日誌。這是通過id關鍵字完成的，該關鍵字可以是單個系統日誌或範圍。可選**occurs**關鍵字指示要呼叫applet必須發生系統日誌的次數（預設值為1）。 可選的**period**關鍵字指示事件必須發生的時間（以秒為單位）。它將小程式呼叫的頻率限制在配置的時間段內最多一次。發生為5，週期為30，表示系統日誌必須在30秒內發生5次，才能觸發事件。如果系統日誌在30秒內出現11次，則僅觸發一次applet。**period**的值0表示未定義任何期間。

可以配置多個系統日誌，但範圍不能重疊。

```
ASA(config-applet)# [no] event syslog id
```

```
ASA(config-applet)# no event syslog id
```

occurs值<*n*>的允許範圍為1到4294967295。 **period**值<seconds>的允許範圍為0到604800。0（零）值表示未配置期間。

系統日誌事件示例

在此示例中，EEM在檢測到記憶體不足塊條件時採取行動。如果可用的1550位元組塊耗盡，它將收集**show blocks pool 1550 dump**並儲存到磁碟。它最多每10分鐘執行一次。

```
event manager applet depletedblock
description "Take a snapshot of block output when it is depleted"
event syslog id 321007 period 600
action 1 cli command "show blocks pool 1550 dump"
output file rotate 10
```

定期事件

EEM還可以配置為定期執行操作。配置基於計時器的事件時，請在事件配置中使用**timer**關鍵字。有3個基於計時器的選項：

- absolute — 第一個計時器是**絕對**計時器，每天在指定時間觸發applet一次並自動重新啟動。
  ```
  ASA(config-applet)# [no] event timer absolute time
  ```

```
ASA(config-applet)# no event timer absolute
```

- 倒計時 — 第二個計時器是**倒計時**器，它觸發小程式一次，除非刪除並重新新增，否則不會重新啟動。

```
ASA(config-applet)# [no] event timer countdown time
```

```
ASA(config-applet)# no event timer countdown
```

- 監視程式 — 第三個計時器是**監視程式**計時器，在每個配置的時間段觸發小程式一次，並自動重新啟動。

```
ASA(config-applet)# [no] event timer watchdog time
```

```
ASA(config-applet)# no event timer watchdog
```

定期事件示例

例如，此事件配置每1分鐘ping 192.168.1.100。這可用於確保VPN隧道即使在空閒流量期間也能保持正常運行狀態。它每60秒使用**監視**程式計時器執行一次。

```
event manager applet period-event
description "Run a command once per minute"
event timer watchdog time 60
action 0 cli command "ping 192.168.1.100"
output none
```

此applet每小時記錄記憶體塊分配資訊，並將輸出寫入一組循環日誌檔案，因為它保留了一天的日誌。它每1小時使用**監**視器計時器執行一次。

```
event manager applet blockcheck
description "Log block usage"
event timer watchdog time 3600
output rotate 24
action 1 cli command "show blocks old"
```

這些小程式會在午夜到凌晨3點之間禁用給定的介面(Gig 0/0)。它使用**絕**對計時器每天執行一次。

```
event manager applet disableintf
description "Disable the interface at midnight"
event timer absolute time 0:00:00
output none
action 1 cli command "interface GigabitEthernet 0/0"
action 2 cli command "shutdown"
action 3 cli command "write memory"
!
event manager applet enableintf
description "Enable the interface at 3am"
event timer absolute time 3:00:00
output none
action 1 cli command "interface GigabitEthernet 0/0"
```

```
action 2 cli command "no shutdown"
action 3 cli command "write memory"
```

手動事件

這些EEM小程式也可以手動呼叫。為此，小程式必須配置**event none**。要手動運行小程式，請輸入**event manager run**命令，後跟小程式的名稱。如果applet配置為除「none」以外的任何事件觸發機制，則嘗試手動運行它將會生成錯誤。使用前面的一個示例「depletedblock」，您可以看到：

```
ASA# event manager run depletedblock
ERROR: Applet not configured with 'event none'
```

手動事件示例

手動事件的使用方式與宏類似。例如，手動事件可用於按順序執行幾個命令。在此範例中，它會儲存組態、對主機執行ping操作，並清除所有回送。

```
event manager applet clean-up
event none
action 0 cli command "write mem"
action 1 cli command "ping 192.168.1.100"
action 2 cli command "clear shun"
output none
```

崩潰事件

當ASA上發生崩潰時，**crashinfo**事件會觸發小程式。無論**output**命令的值如何，**action**命令都會定向到crashinfo檔案。在生成crashinfo的**show tech**部分之前生成輸出。

> **警告：**當ASA崩潰時，機箱的狀態通常未知。當裝置處於此狀態時，某些CLI命令可能無法安全運行。

```
ASA(config-applet)# [no] event crashinfo
```

操作配置

當觸發小程式時，將執行小程式上的操作。每個**操作**都有一個序號，用於指定操作的順序。每個小程式可以配置多個操作；但每個序數只能使用一次。這些命令是典型的CLI命令，例如**show blocks**。強烈建議使用報價，但不需要報價。

```
ASA(config-applet)# [no] action
```

```
ASA(config-applet)# no action
```

操作識別符號*<n>的值範圍*為0到4294967295。*<command>*的值必須用引號，否則如果命令由多個字組成，則會發生錯誤。該命令在配置模式下以許可權級別為15（最高）的使用者身份執行。 該命令可能不接受任何輸入；如果命令具有**noconfirm**選項，則as input將被禁用。由於命令不會以互動方式處理，因此應使用此引數。

輸出配置

這些操作的輸出可以通過**output**命令定向到指定的位置。每次只能啟用一個輸出值。預設值為**output none**。此值會丟棄action命令的所有輸出。

```
ASA(config-applet)# [no] output none
```

output console命令將action命令的輸出傳送到控制檯。

```
ASA(config-applet)# [no] output console
```

output file命令將操作命令的輸出定向到檔案。可以使用四個選項。**new**選項為每個呼叫將applet的輸出寫入新檔案。*filename*的格式為**eem-*<applet>*-*<timestamp>*.log**。其中*<applet>是*小程式的名稱，*<timestamp>*是*YYYYMMDD-hhmmss格式的日期時間戳*。

```
ASA(config-applet)# [no] output file new
```

rotate選項用於建立一組像Linux的日誌旋轉機制旋轉的檔案。檔名格式為**eem-*<applet>*-*<x>*.log**。其中*<applet>*是該applet的名稱，*<x>是*檔案編號。最新的檔案用數字0（零）表示，而最舊的檔案用最高數字(*<n>*-1)表示。 寫入新檔案時，最舊的檔案會被刪除，並且所有後續檔案都會在寫入第0個檔案之前重新編號。

```
ASA(config-applet)# [no] output file rotate
```

旋轉值*<n>*的範圍為2到100。

overwrite選項用於始終將action命令輸出寫入每次被截斷的單個檔案。

```
ASA(config-applet)# [no] output file overwrite
```

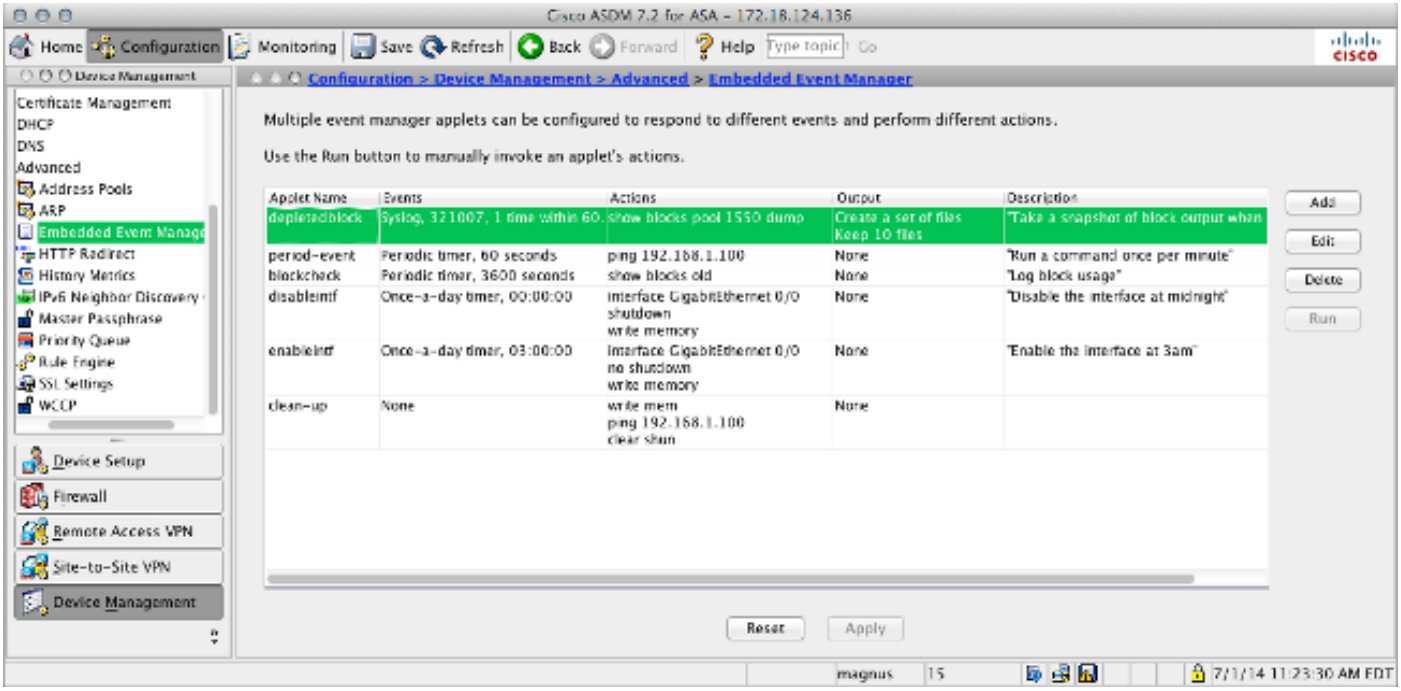
append選項始終用於將操作命令輸出寫入單個檔案，但該檔案每次都會被附加。

```
ASA(config-applet)# [no] output file append
```

*<filename>*引數是本地(ASA)檔名。overwrite命令也可能使用**ftp:**、**tftp:**和**smb:**目標檔案。

ASDM配置

也可以從ASDM內部配置EEM。選擇**Configuration > Device Management > Advanced > Embedded Event Manager**。在ASDM的這一部分，您可以使用前面討論的相同引數配置EEM小程式。配置applet後，按一下**Apply**將配置推送到ASA。



驗證

執行模式命令

使用本節內容，確認您的組態是否正常運作。

所有這些命令均在執行模式下使用。

此命令顯示事件管理器系統的運行配置。

```
ASA# show running-config event manager
```

此命令執行已配置了**event none**的事件管理器小程式。如果運行未使用**event none**配置的小程式，則會報告錯誤。

```
ASA# event manager run
```

```
      appletappletASA# event manager applet period-event, hits 1, last 2014/07/01 10:51:52
last file none
```

event watchdog 60 secs, left 54 secs, hits 1, last 2014/07/01 10:51:52
action 0 cli command "ping 192.168.1.100", hits 1, last 2014/07/01 10:51:52show counter CLIeem

ASA# show counters protocol eem ()showshow

EEM debug Debug ASA# [no] debug event manager

ASA# show debug event manager