システム イベントに対する応答の自動化

この章では、Embedded Event Manager（EEM）を設定する方法について説明します。

EEM について

EEM サービスを利用することで、問題をデバッグし、トラブルシューティングに対して汎用ロギングを提供できます。EEM サービスには 2 つのコンポーネント、つまり EEM が応答またはリッスンするイベント、およびアクションと EEM が応答するイベントを定義するイベント マネージャ アプレットがあります。さまざまなイベントに応答し、さまざまなアクションを実行するために、複数のイベント マネージャ アプレットを設定できます。

サポートされるイベント

EEM は次のイベントをサポートします。

- Syslog：ASA は、syslog メッセージの ID を使用して、イベント マネージャ アプレットをトリガーする syslog メッセージを識別します。複数の syslog イベントを設定できますが、単一のイベント マネージャ アプレット内で syslog メッセージの ID が重複することはできません。
- タイマー：タイマーを使用して、イベントをトリガーできます。各タイマーは、各イベント マネージャ アプレットに対して一度だけ設定できます。各イベント マネージャ アプレットには最大で 3 つのタイマーがあります。3 種類のタイマーは次のとおりです。
  - ウォッチドッグ（定期的）タイマーは、アプレットアクションの完了後に指定された期間が経過するとイベント マネージャ アプレットをトリガーし、自動的にリスタートします。

- カウントダウン（ワンショット）タイマーは、指定された期間が経過するとイベントマネージャアプレットを1回トリガーします。削除および再追加されない限りはリスタートしません。
- 絶対（1 日 1 回）タイマーは、イベントを 1 日 1 回指定された時刻に発生させ、自動的にリスタートします。時刻の形式は hh:mm:ss です。

  各イベント マネージャ アプレットに対して、各タイプのタイマー イベントを 1 つだけ設定できます。

- なし：CLI または ASDM を使用してイベント マネージャ アプレットを手動で実行する場合、イベントはトリガーされません。
- クラッシュ：ASA がクラッシュした場合、クラッシュ イベントがトリガーされます。**output** コマンドの値に関係なく、**action** コマンドはクラッシュ情報ファイルを対象とします。出力は **show tech** コマンドの前に生成されます。

イベント マネージャ アプレットのアクション

イベント マネージャ アプレットがトリガーされると、そのイベント マネージャ アプレットのアクションが実行されます。各アクションには、アクションの順序を指定するために使用される番号があります。このシーケンス番号は、イベント マネージャ アプレット内で一意である必要があります。イベント マネージャ アプレットには複数のアクションを設定できます。コマンドは典型的な CLI コマンドです（**show blocks** など）。

出力先

output コマンドを使用すると、アクションの出力を指定した場所に送信できます。一度にイネーブルにできる出力値は1つだけです。デフォルト値は **output none** です。この値は、**action** コマンドによるすべての出力を破棄します。このコマンドは、特権レベル 15（最高）を持つユーザーとして、グローバル コンフィギュレーション モードで実行されます。ディセーブルになっているため、このコマンドは入力を受け付けない場合があります。次の3つの場所のいずれかに **action** CLI コマンドの出力を送信できます。

- **なし**：デフォルトの設定です。出力を破棄します。
- **コンソール**：出力を ASA コンソールに送信します。
- **ファイル**：出力をファイルに送信します。次の 4 つのファイル オプションを使用できます。
  - **一意のファイルを作成する**：イベントマネージャアプレットが呼び出されるたびに、一意の名前を持つ新しいファイルを作成します。
  - **ファイルを作成する/ファイルを上書きする**：イベント マネージャ アプレットが呼び出されるたびに、指定されたファイルを上書きします。

- **ファイルを作成する/ファイルに付加する**：イベント マネージャ アプレットが呼び出されるたびに、指定されたファイルに付加します。ファイルがまだ存在しない場合は作成されます。
- **一連のファイルを作成する**：イベント マネージャ アプレットが呼び出されるたびにローテーションされる、一意の名前を持つ一連のファイルを作成します。

EEM のガイドライン

ここでは、EEM を設定する前に確認する必要のあるガイドラインおよび制限事項について説明します。

コンテキスト モードのガイドライン

マルチ コンテキスト モードではサポートされません。

その他のガイドライン

- 通常、クラッシュ時は、ASA の状態は不明です。こうした状況では、一部のコマンドの実行は安全ではない可能性があります。
- イベント マネージャ アプレットの名前にはスペースを含めることができません。
- None イベントおよび Crashinfo イベント パラメータは変更できません。
- syslog メッセージが EEM に送信されて処理されるため、パフォーマンスが影響を受ける可能性があります。
- 各イベント マネージャ アプレットのデフォルトの出力は **output none** です。この設定を変更するには、異なる出力値を入力する必要があります。
- 各イベント マネージャ アプレットに定義できる出力オプションは 1 つだけです。

EEM の設定

EEM の設定は、次のタスクで構成されています。

手順

ステップ 1 イベント マネージャ アプレットの作成とイベントの設定 （4 ページ）。
ステップ 2 アクションおよびアクションの出力先の設定 （6 ページ）を使用して無効にすることができます。
ステップ 3 イベント マネージャ アプレットの実行 （8 ページ）を使用して無効にすることができます。

ステップ 4 トラック メモリ割り当ておよびメモリ使用量 （8 ページ）を使用して無効にすることができます。

イベント マネージャ アプレットの作成とイベントの設定

イベントマネージャアプレットを作成してイベントを設定するには、次の手順を実行します。

手順

ステップ 1 イベント マネージャ アプレットを作成し、イベント マネージャ アプレットのコンフィギュレーション モードを開始します。

event manager applet *name*

例：

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

name 引数には、最大 32 文字の英数字を指定できます。スペースは使用できません。

イベント マネージャ アプレットを削除するには、このコマンドを **no** 形式で入力します。

ステップ 2 イベント マネージャ アプレットの説明を入力します。

description *text*

例：

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

text 引数は、最大 256 文字です。引用符内であれば、説明テキストにスペースを含めることができます。

ステップ 3 指定されたイベントを設定するには、次のコマンドのいずれかを入力します。設定されたイベントを削除するには、それぞれのコマンドを **no** 形式で入力します。

- syslog イベントを設定するには、イベント マネージャ アプレットをトリガーする単一の syslog メッセージまたは syslog メッセージの範囲を指定します。

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

  例：

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

  nnnnnn 引数には、syslog メッセージの ID を指定します。キーワードと引数のペアである **occurs** *n* は、イベント マネージャ アプレットを呼び出すために syslog メッセージが発生しなければならない回数を示しています。デフォルトの発生回数は 0 秒ごとに 1 回です。

有効な値は、1 ～ 4294967295 です。キーワードと引数のペアである **period** *seconds* は、イベントが発生する際の許容時間（秒数）を示しています。また、イベント マネージャ アプレットが設定された期間に 1 回呼び出される際の最大の間隔を制限します。有効な値は、0 ～ 604800 です。値 0 は、期間が定義されていないことを示しています。

- イベントを設定された期間ごとに1回発生させ、自動的にリスタートするように設定します。

  event timer watchdog time *seconds*

  例：

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

  秒数は、1 ～ 604800 の範囲で設定してください。

- イベントを1回発生させ、削除および再追加されない限りはリスタートしないように設定します。

  event timer countdown time *seconds*

  例：

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

  秒数は、1 ～ 604800 の範囲で設定してください。カウントダウン タイマー イベントを削除するには、このコマンドの **no** 形式を使用します。

  （注） スタートアップ コンフィギュレーションである場合、このタイマーはリブート時に再実行されます。

- イベントを 1 日 1 回指定された時刻に発生させ、自動的にリスタートするように設定します。

  event timer absolute time *hh:mm:ss*

  例：

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

  時刻の形式は hh:mm:ss です。時刻の範囲は 00:00:00（真夜中）から 23:59:59 です。

- ASA のクラッシュ時にクラッシュ イベントをトリガーします。

  event crashinfo

  例：

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

output コマンドの値に関係なく、**action** コマンドはクラッシュ情報ファイルを対象とします。出力は **show tech** コマンドの前に生成されます。

アクションおよびアクションの出力先の設定

アクションおよびアクションの出力を送信する特定の宛先を設定するには、次の手順を実行します。

手順

ステップ 1 イベント マネージャ アプレットにアクションを設定します。

action *n* **cli command** *"command"*

例：

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

n オプションはアクション ID です。有効な ID の範囲は、0 ～ 4294967295 です。*command* オプションの値は、引用符で囲む必要があります。引用符で囲んでいない場合、コマンドが 2 つ以上の単語で構成されているとエラーが発生します。このコマンドは、特権レベル 15（最高）を持つユーザーとして、グローバル コンフィギュレーション モードで実行されます。ディセーブルになっているため、このコマンドは入力を受け付けない場合があります。コマンドで使用可能な場合は、**noconfirm** オプションを使用します。

ステップ 2 使用可能な出力先オプションを 1 つ選択します。出力先を削除するには、各コマンドの **no** 形式を使用します。

- **None** オプションは、**action** コマンドからのあらゆる出力を破棄します。これがデフォルト設定です。

  output none

  例：

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

- **Console** オプションは、**action** コマンドの出力をコンソールに送信します。

  output console

  例：

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

  （注） このコマンドを実行すると、パフォーマンスに影響を及ぼします。

- **New File** オプションは、呼び出された各イベント マネージャ アプレットの新しいファイルに **action** コマンドの出力を送信します。

  output file new

  例：

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

  ファイル名の形式は、eem-*applet-timestamp*.log です。ここで、*applet* はイベント マネージャ アプレットの名前、*timestamp* は日付のタイム スタンプ（形式は YYYYMMDD-hhmmss）を示しています。

- **New Set of Rotated Files** オプションは、ローテーションされる一連のファイルを作成します。新しいファイルが書き込まれる場合、最も古いファイルが削除され、最初のファイルが書き込まれる前に後続のすべてのファイルに番号が再度割り振られます。

  output file rotate *n*

  例：

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

  最も新しいファイルが 0 で示され、最も古いファイルが最大数（*n* -1）で示されます。*n* オプションはローテーションの値です。有効な値の範囲は 2 ～ 100 です。ファイル名の形式は、eem-*applet-x*.log です。ここで、*applet* はアプレットの名前、*x* はファイル番号を示しています。

- **Single Overwritten File** オプションは、**action** コマンドの出力を単一のファイルに書き込みます。このファイルは毎回上書きされます。

  output file overwrite *filename*

  例：

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

  filename 引数は、（ASA に対して）ローカルのファイル名です。このコマンドは、FTP、TFTP、および SMB のターゲット ファイルを使用する場合もあります。

- **Single Appended File** オプションは、**action** コマンドの出力を単一のファイルに書き込みますが、このファイルは毎回上書きされます。

  output file append *filename*

  例：

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

filename 引数は、（ASA に対して）ローカルのファイル名です。

イベント マネージャ アプレットの実行

イベント マネージャ アプレットを実行するには、次の手順を実行します。

手順

イベント マネージャ アプレットを実行します。

event manager run *applet*

例：

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

event none コマンドで設定されていないイベント マネージャ アプレットを実行すると、エラーが発生します。*applet*引数は、イベント マネージャ アプレットの名前です。

トラック メモリ割り当ておよびメモリ使用量

メモリ割り当てとメモリ使用量をログに記録するには、次の手順を実行します。

手順

ステップ 1 メモリ ロギングをイネーブルにします。

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

例：

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

必要な唯一の引数は、メモリ ロギング バッファ内のエントリ数です。**wrap** オプションは、ラップ時にバッファを保存するようメモリ ロギング ユーティリティに指示します。保存できるのは一度だけです。

メモリ ロギング バッファが複数回ラップした場合は、上書きされます。バッファがラップすると、そのデータの保存をイネーブルにするトリガーがイベント マネージャに送信されます。**size** オプションは、特定のサイズをモニターします。**process** オプションは、特定のプロセスをモニタします。

（注） Checkheaps プロセスは、非標準の方法でメモリ アロケータを使用するため、プロセスとして完全に無視されます。

context オプションは、指定した名前で特定の仮想コンテキストのメモリ ロギングを記録します。

メモリ ロギング パラメータを変更するには、それをディセーブルにしてから、再度イネーブルにします。

ステップ 2 メモリ ロギング結果を表示します。

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

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 0x000000000013193e8
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)
```

どのオプションも指定しない場合、**show memory logging** は統計情報を表示し、記録された処理を表示します。**brief** オプションは、統計情報だけを表示します。**wrap** オプションは、重複したデータが表示または保存されないように、ラップ時点でバッファを表示してから、そのデータを消去します。**include** オプションは、指定されたフィールドのみを出力に含めます。任意の順序でフィールドを指定できますが、必ず次の順序で表示されます。

1. プロセス
2. 時刻
3. コンテキスト（シングル モード以外）
4. 処理（free/malloc/など）

5. アドレス
6. サイズ
7. 発信者

出力形式は、次のとおりです。

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

最大4つの発信者アドレスが表示されます。例に示すように、処理の種類（番号）が出力に列挙されます。

ステップ3 メモリ ロギング ラップ イベントに応答します。

event memory-logging-wrap

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
```

この例では、すべてのメモリ割り当てを記録するアプレットを示します。メモリ ロギングに対してラップがイネーブルになっている場合は、メモリ ロガーが、設定されたアプレットをトリガーするイベントをイベント マネージャに送信します。

EEM の例

次に、ブロックの漏えい情報を1時間ごとに記録し、その出力をローテーションされる一連のログ ファイルに書き込み、1 日分のログを保持するイベント マネージャ アプレットの例を示します。

```
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"
```

次に、毎日午前 1 時に ASA をリブートし、必要に応じて設定を保存するイベント マネージャ アプレットの例を示します。

```
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"
```

次に、午前 0 時から午前 3 時の間に特定のインターフェイスをディセーブルにするイベント マネージャ アプレットの例を示します。

```
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"
```

EEM のモニタリング

EEM をモニターするには、次のコマンドを参照してください。

- **clear configure event manager**

  このコマンドは、イベント マネージャの実行コンフィギュレーションを削除します。

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

  このコマンドは、コンフィギュレーションから指定のイベント マネージャ アプレットを削除します。

- **show counters protocol eem**

  このコマンドは、イベント マネージャのカウンタを表示します。

- **show event manager**

  このコマンドは、ヒット カウントやイベント マネージャ アプレットが最後に呼び出されたのはいつかなど、設定されたイベント マネージャ アプレットに関する情報を表示します。

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

  これらのコマンドは、メモリ割り当てとメモリ使用量に関する統計情報を表示します。

- **show running-config event manager**

  このコマンドは、イベント マネージャの実行コンフィギュレーションを表示します。

EEM の履歴

表 *1 : EEM* の履歴

機能名	プラットフォームリリース	説明
Embedded Event Manager（EEM）	9.2(1)	EEMサービスを利用することで、問題をデバッグし、トラブルシューティングに対して汎用ロギングを提供できます。EEMサービスには2つのコンポーネント、つまりEEMが応答またはリッスンするイベント、およびアクションと EEM が応答するイベントを定義するイベント マネージャ アプレットがあります。さまざまなイベントに応答し、さまざまなアクションを実行するために、複数のイベント マネージャ アプレットを設定できます。 **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** の各コマンドが導入または変更されました。
EEM のメモリ トラッキング	9.4(1)	メモリ割り当てとメモリ使用量をログに記録し、メモリ ロギング ラップ イベントに応答する新しいデバッグ機能が追加されました。 **memory logging**、**show memory logging**、**show memory logging include**、**event memory-logging-wrap** の各コマンドが導入または変更されました。

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。