ASA Embedded Event Manager 컨피그레이션 예

목차

소개

이 문서에서는 ASA(Adaptive Security Appliance) 버전 9.2(1)에 추가된 문제 해결 툴인 EEM(Embedded Event Manager)에 대해 설명합니다. 이 기능은 Cisco IOS와 비슷합니까? 기반 EEM ASA 이벤트(syslogs)를 기반으로 CLI 명령을 실행하고 출력을 저장하는 강력한 방법입니다. 이 문서에서는 기능에 대한 소개와 일부 EEM 애플릿에 대해 설명합니다.

사전 요구 사항

요구 사항

EEM을 사용하려면 ASA가 단일 컨텍스트 모드로 구성되어 있어야 합니다.

사용되는 구성 요소

이 문서의 정보는 ASA 버전 9.2(1) 이상을 기반으로 합니다.

지침 및 제한 사항

이 섹션에는 이 기능에 대한 지침 및 제한 사항이 포함되어 있습니다.

컨텍스트 모드 지침

EEM은 현재 단일 컨텍스트 모드에서 실행되는 ASA 방화벽에서만 지원됩니다. 다중 컨텍스트 모드로 구성된 방화벽은 현재 지원되지 않습니다.

방화벽 모드 지침

EEM은 현재 라우팅 및 투명 방화벽 모드에서 모두 지원됩니다.

추가 지침

- 유닛이 crash하는 동안 ASA의 상태를 일반적으로 알 수 없습니다. ASA가 이 상태에서는 일부 명령을 실행하는 것이 안전하지 않을 수 있습니다.
- 이벤트 관리자 애플릿의 이름에는 공백을 포함할 수 없습니다.
- None 이벤트 및 Crashinfo 이벤트 매개변수는 수정할 수 없습니다.
- syslog 메시지가 EEM으로 전송되어 처리되므로 성능이 저하될 수 있습니다.
- 기본 출력은 각 이벤트 관리자 애플릿에 대해 **output none**입니다. 기본 출력을 변경하려면 다른 출력 값을 입력해야 합니다.
- 각 이벤트 관리자 애플릿에 대해 하나의 출력 옵션만 정의할 수 있습니다.

구성

event manager applet 명령은 이벤트를 작업 및 출력과 연결하는 프로세스인 이벤트 관리자 애플릿을 생성/편집합니다. *<name>*은 32자로 제한되며 공백을 포함할 수 없습니다. 이벤트 관리자 애플릿 하위 모드를 시작합니다.

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

설명을 애플릿에 추가할 수 있습니다. 이는 정보 제공 목적으로만 사용됩니다. <*text*>은(는) 256자로 제한됩니다.

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

이벤트 구성

애플릿을 트리거하는 애플릿에 구성된 작업을 호출하기 위한 다양한 이벤트가 추가될 수 있습니다. **event** 키워드로 정의됩니다. 각 애플릿에 대해 여러 이벤트가 구성될 수 있습니다.

Syslog 이벤트

지원되는 첫 번째 이벤트 유형은 syslog**입니다**. ASA는 애플릿을 트리거하는 syslog를 식별하기 위해 syslog ID를 사용합니다. 이는 단일 syslog 또는 범위일 수 있는 id 키워드를 통해 완료됩니다. 선택적 **occurs** 키워드는 애플릿을 호출하기 위해 syslog가 발생해야 하는 횟수를 나타냅니다(기본값은 1). 선택적 **period** 키워드는 이벤트가 발생해야 하는 시간(초)을 나타냅니다. 구성된 기간 동안 애플릿 호출의 빈도를 최대 한 번으로 제한합니다. 은 30 **기간**으로 5**일** 경우, 이벤트가 트리거되기 전 30초 이내에 syslog가 5회 발생해야 함을 의미합니다. syslog가 30초 내에 11번 발생하면 애플릿은 한 번만 트리거됩니다. **기간**에 대한 값 0은 기간이 정의되지 않았음을 의미합니다.

여러 syslog를 구성할 수 있지만 범위는 겹칠 수 없습니다.

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

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

occurs 값 *<n>*의 허용 범위는 1~4294967295입니다. **기간** 값 <seconds>의 허용 범위는 0~604800입니다. 0(영) 값은 구성된 기간이 없음을 의미합니다.

Syslog 이벤트 예

이 예에서 EEM은 메모리 부족 블록 조건을 탐지하면 조치를 취합니다. 사용 가능한 1550바이트 블록이 고갈되면 **show blocks pool 1550 dump**를 **수집하고** 디스크에 저장합니다. 10분마다 한 번씩 이런 일을 합니다.

```
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
```

정기 이벤트

EEM은 주기적으로 작업을 수행하도록 구성할 수도 있습니다. 타이머 기반 이벤트를 구성할 때 이벤트 컨피그레이션에서 **timer** 키워드를 사용합니다. 3가지 타이머 기반 옵션이 있습니다.

- absolute - 첫 번째 타이머는 지정된 시간에 하루에 한 번 애플릿을 트리거하고 자동으로 재시작하는 **절대** 타이머입니다.
  ```
  ASA(config-applet)# [no] event timer absolute time
  ```

  ```
  ASA(config-applet)# no event timer absolute
  ```
- countdown - 두 번째 타이머는 **카운트다운** 타이머로 애플릿을 한 번 트리거하고 제거한 후 다시 추가하지 않으면 다시 시작되지 않습니다.
  ```
  ASA(config-applet)# [no] event timer countdown time
  ```

  ```
  ASA(config-applet)# no event timer countdown
  ```
- watchdog - 세 번째 타이머는 **watchdog** 타이머로 구성된 기간마다 한 번 애플릿을 트리거하고 자동으로 재시작합니다.
  ```
  ASA(config-applet)# [no] event timer watchdog time
  ```

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

정기 이벤트 예

예를 들어, 이 이벤트 컨피그레이션 ping은 1분마다 192.168.1.100. 이는 유휴 트래픽이 있는 동안에도 VPN 터널이 작동 및 작동되도록 하는 데 사용될 수 있습니다. 60초마다 **워치독** 타이머를 사용합니다.

```
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
```

이 애플릿은 하루 단위 로그를 보관하므로 매 시간마다 메모리 블록 할당 정보를 기록하고 회전 로그 파일 집합에 출력을 기록합니다. 워치독 타이머를 사용하여 1시간마다 실행됩니다.

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

이 애플릿은 자정부터 오전 3시까지 지정된 인터페이스(Gig 0/0)를 비활성화합니다. **절대** 타이머를 사용하여 하루에 한 번 실행합니다.

```
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"
```

수동 이벤트

이러한 EEM 애플릿은 수동으로 호출할 수도 있습니다. 이 작업을 수행하려면 애플릿에서 **이벤트 없음**을 구성해야 합니다. 애플릿을 수동으로 실행하려면 **event manager run** 명령 뒤에 애플릿의 이름을 입력합니다. 애플릿이 'none' 이외의 이벤트 트리거 메커니즘에 대해 구성된 경우 애플릿을 수동으로 실행하려고 하면 오류가 발생합니다. 이전 예인 'deployedblock'을 사용하면 다음을 확인할 수 있습니다.

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

수동 이벤트 예

수동 이벤트는 매크로와 유사한 방식으로 사용할 수 있습니다. 예를 들어 수동 이벤트를 사용하여 몇 가지 명령을 순서대로 실행할 수 있습니다. 이 예에서는 컨피그레이션을 저장하고 호스트를 ping하고 모든 shun을 지웁니다.

```
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
```

충돌 이벤트

crashinfo 이벤트는 ASA에서 충돌이 발생할 때 애플릿을 트리거합니다. **output** 명령의 값에 관계없이 **action** 명령은 crashinfo 파일로 전달됩니다. crashinfo의 **show tech** 부분이 생성되기 전에 출력이 생성됩니다.

> **경고:** ASA가 다운되면 일반적으로 상자의 상태를 알 수 없습니다. 디바이스가 이 상태이면 일부 CLI 명령을 실행하는 것이 안전하지 않을 수 있습니다.

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

작업 구성

애플릿이 트리거되면 애플릿에 대한 작업이 수행됩니다. 각 **작업**에는 작업의 순서를 지정하는 데 사용되는 서수가 있습니다. 애플릿당 여러 작업을 구성할 수 있습니다. 각 서수는 한 번만 사용할 수 있습니다. 명령은 **show blocks**와 같은 일반적인 CLI 명령입니다. 견적은 권장되지만 필수는 아닙니다.

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

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

작업 식별자 값 *<n>*의 범위는 0에서 4294967295까지입니다. *<command>* 값은 따옴표로 묶어야 합니다. 그렇지 않으면 명령이 두 개 이상의 단어로 구성된 경우 오류가 발생합니다. 이 명령은 권한 수준 15(최고)의 사용자로 컨피그레이션 모드에서 실행됩니다. 명령은 어떤 입력도 허용하지 않을 수 있습니다. 명령에 noconfirm 옵션이 있는 경우 input이 **비활성화됩니다**. 명령은 대화식으로 처리되지 않으므로 사용해야 합니다.

출력 구성

작업의 출력은 **output** 명령을 통해 지정된 위치로 **전달됩** 수 있습니다. 한 번에 하나의 출력 값만 활성화할 수 있습니다. 기본값은 output none**입니다**. 이 값은 action 명령의 모든 출력을 무시합니다.

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

output **console** 명령은 action 명령의 출력을 콘솔로 전송합니다.

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

output **file** 명령은 action 명령의 출력을 파일로 전달합니다. 사용 가능한 4가지 옵션이 있습니다. **new** 옵션은 각 호출의 새 파일에 애플릿의 출력을 기록합니다. 파일 *이름*의 형식은 eem-***<applet>-<timestamp>*.log입니다**. 여기서 *<applet>*은 애플릿의 이름이고 *<timestamp>*는 YYYYYMMDD-hhmmss 형식의 날짜 타임스탬프입니다.

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

rotate 옵션은 Linux의 로그 회전 메커니즘과 비슷하게 순환되는 파일 집합을 만드는 데 사용됩니다. 파일 이름 형식은 **eem-*<applet>*-*<x>*.log입니다**. 여기서 *<applet>*은 애플릿의 이름이고 *<x>* 은 파일 번호입니다. 최신 파일은 숫자 0(0)으로 표시되고 가장 오래된 파일은 가장 높은 숫자(*<n>*-1)로 표시됩니다. 새 파일을 작성할 때 가장 오래된 파일이 삭제되고 이후의 모든 파일은 10번째 파일이 기록되기 전에 다시 번호가 매겨집니다.

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

회전 값 *<n>*의 범위는 2에서 100입니다.

overwrite 옵션은 작업 명령 출력을 항상 잘리는 단일 파일에 쓰는 데 사용됩니다.

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

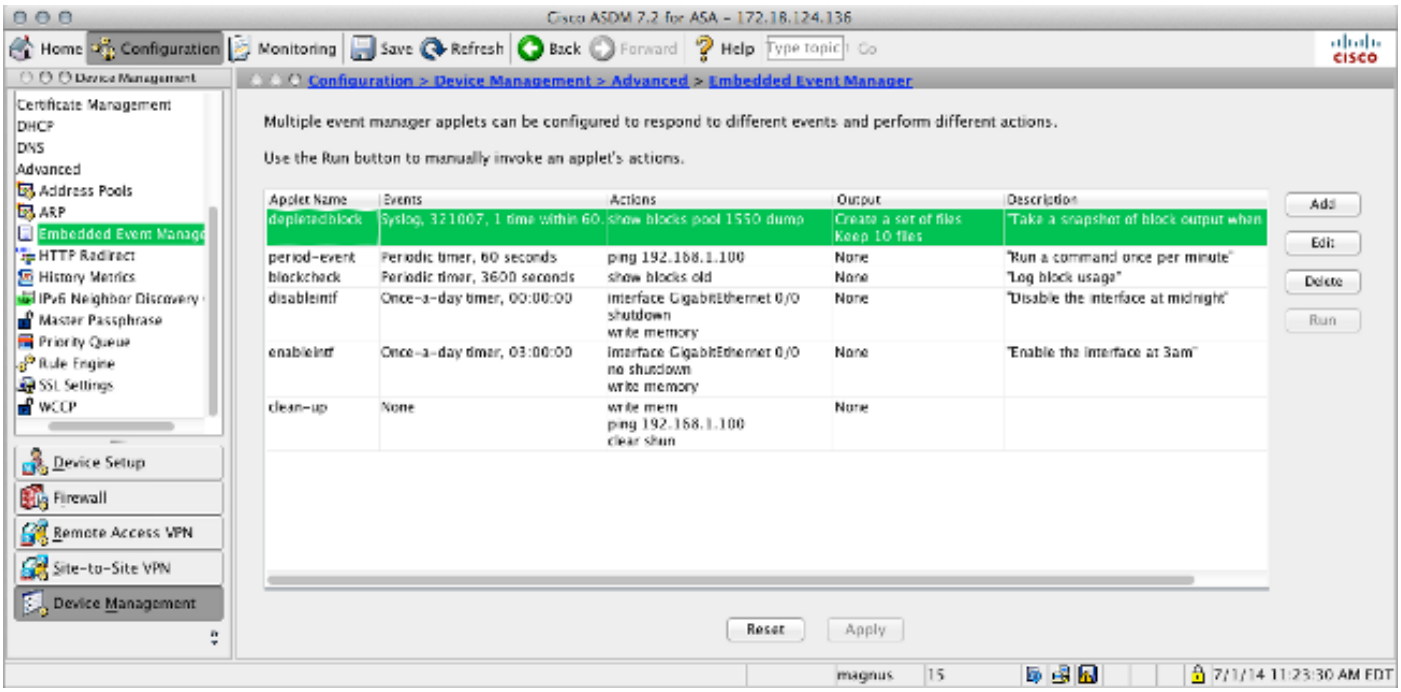
append 옵션은 action 명령 출력을 항상 단일 파일에 쓰는 데 사용되지만, 이 파일은 매번 추가됩니다.

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

<filename> 인수는 ASA에 대한 로컬 파일 이름입니다. overwrite 명령은 **ftp:**, **tftp:** 및 **smb:** 대상 파일.

ASDM 컨피그레이션

ASDM 내에서 EEM을 구성할 수도 있습니다. Configuration **> Device Management > Advanced > Embedded Event Manager를 선택합니다**. ASDM의 이 섹션에서는 앞서 설명한 것과 동일한 매개변수로 EEM 애플릿을 구성할 수 있습니다. 애플릿을 구성한 후 **Apply(적용)**를 클릭하여 컨피그레이션을 ASA에 푸시합니다.



다음을 확인합니다.

실행 모드 명령

이 섹션을 사용하여 컨피그레이션이 제대로 작동하는지 확인합니다.

이 모든 명령은 실행 모드에서 사용됩니다.

이 명령은 이벤트 관리자 시스템의 실행 중인 컨피그레이션을 보여줍니다.

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

이 명령은 **이벤트 없음**으로 구성된 이벤트 관리자 애플릿을 **실행합니다**. **이벤트 없음**으로 구성되지 않은 애플릿을 실행하는 경우 오류가 보고됩니다.

```
ASA# event manager run
```

```
            .          .ASA# 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 10:51:52    .show counter CLI
  eem     .
```

```
   ASA# show counters protocol eem Output Interpreter (  ) show  . show     [  ] .
```

```
   EEM      .: debug     ____ . ASA# [no] debug event manager
```

```
ASA# show debug event manager            .              .
```