EEM의 모범 사례 및 유용한 스크립트 이해

목차

소개

이 문서에서는 Cisco IOS® XE 장치에 대한 EEM(Embedded Event Manager) 스크립트 구성 모범 사례에 대해 설명합니다.

사전 요구 사항

요구 사항

Cisco에서는 이 주제에 대해 잘 알고 숙지할 것을 권장합니다.

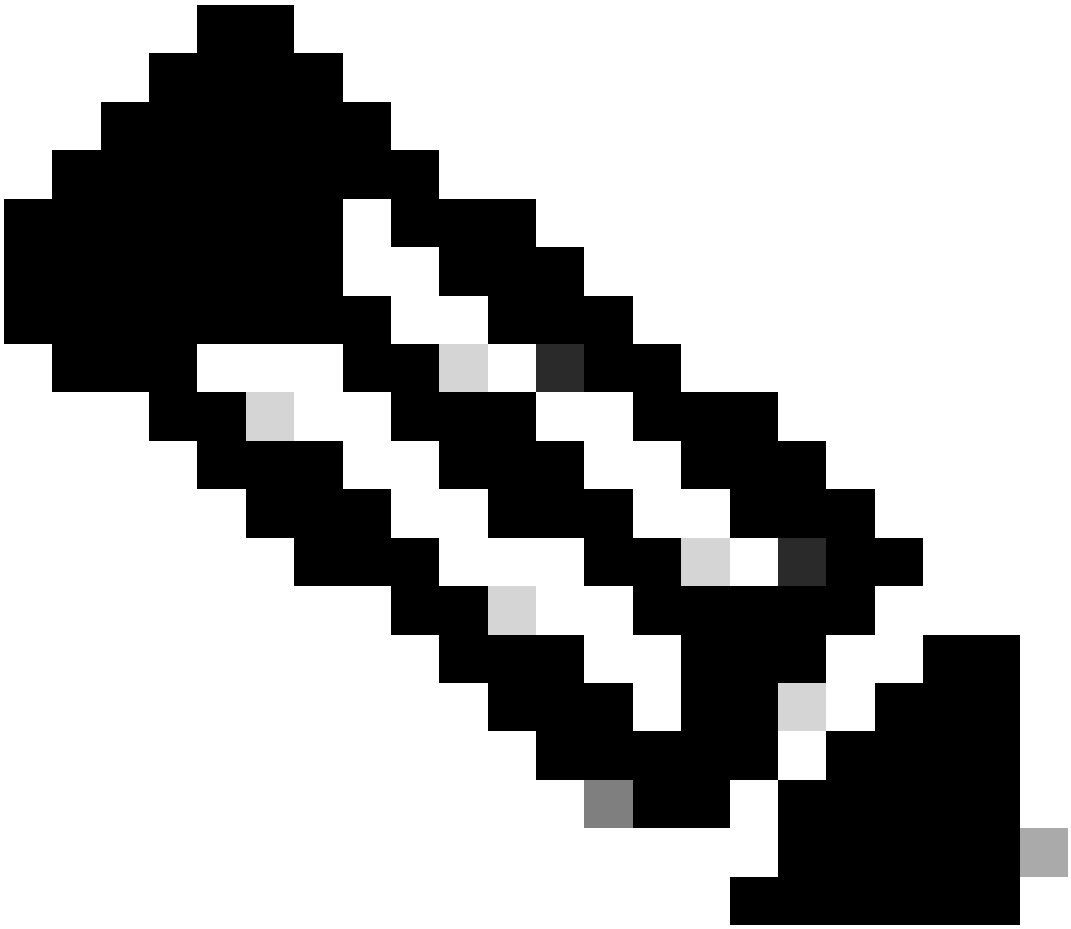
- Cisco IOS 및 Cisco IOS XE EEM(Embedded Event Manager)

이 기능에 대해 잘 모르는 경우 먼저 EEM 기능 개요를 읽어 보십시오.

사용되는 구성 요소

이 문서의 정보는 다음 소프트웨어 및 하드웨어 버전을 기반으로 합니다.

- Cisco Catalyst 9300, 9400 및 9500 스위치
- Cisco IOS Software 버전 16.X 또는 17.X



참고: 이 스크립트는 Cisco TAC에서 지원하지 않으며 교육 목적으로 있는 그대로 제공됩니다.

이 문서의 정보는 특정 랩 환경의 디바이스를 토대로 작성되었습니다. 이 문서에 사용된 모든 디바이스는 초기화된(기본) 컨피그레이션으로 시작되었습니다. 현재 네트워크가 작동 중인 경우 모든 명령의 잠재적인 영향을 미리 숙지하시기 바랍니다.

표기 규칙

문서 규칙에 대한 자세한 내용은 Cisco 기술 팁 표기 규칙을 참조하십시오.

모범 사례

이 섹션에서는 EEM 스크립트의 설계 및 구현과 관련된 가장 일반적인 몇 가지 문제를 다룹니다. EEM 모범 사례에 대한 자세한 내용은 참조 섹션에서 참조하는 EEM 모범 사례 문서를 참조하십시오.

적절한 인증이 사용되고 있는지 확인합니다.

디바이스에서 AAA를 사용하는 경우, 디바이스에 구성된 EEM 스크립트가 스크립트에서 명령을 실행할 수 있는 AAA 사용자로 구성되었는지 또는 권한 부여 우회가 스크립트 정의의 명령 권한 부여 우회로 구성되었는지 확인해야 합니다.

EEM 런타임 및 속도 제한에 대한 제약 조건 추가

기본적으로 EEM 스크립트는 최대 20초 동안 실행할 수 있습니다. 실행하는 데 시간이 더 걸리거나 명령 실행 간에 기다려야 하는 스크립트를 디자인하는 경우 애플릿 이벤트 트리거에서 maxrun 값을 지정하여 기본 실행 타이머를 변경합니다.

또한 EEM 스크립트를 트리거하는 이벤트를 실행할 수 있는 빈도를 고려해야 합니다. 짧은 시간 내에 빠르게 발생하는 조건(예: MAC 플랩의 경우 syslog 트리거)에서 스크립트를 트리거하는 경우, EEM 스크립트에 속도 제한 조건을 포함하여 동시에 과도한 실행 수를 방지하고 디바이스 리소스 소모를 방지하는 것이 중요합니다.

잘못된 실행 방지

EEM 문서에 설명된 대로, 작업 명령문의 실행 순서는 레이블에 의해 제어됩니다(예: action 0001 cli command enable의 레이블은 0001). 이 레이블 값은 숫자가 아니라 영숫자입니다. 작업은 오름차순 영숫자 키 순서로 정렬되고 label 인수를 정렬 키로 사용하며 이 시퀀스에서 실행됩니다. 이렇게 하면 작업 레이블을 구성하는 방법에 따라 예기치 않은 실행 순서가 발생할 수 있습니다.

다음 예를 고려하십시오.

```
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"
```

영숫자 비교에서 120은 13 이전이므로 이 스크립트는 예상한 순서대로 실행되지 않습니다. 이를 방지하기 위해 다음과 같은 패딩 시스템을 사용하는 것이 유용합니다.

```
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"
```

여기서의 패딩으로 인해 번호가 매겨진 문장은 예상 순서로 평가됩니다. 각 레이블 사이에 10씩 증가하면 모든 후속 문의 번호를 다시 지정할 필요 없이 나중에 필요한 경우 EEM 스크립트에 추가 문을 삽입할 수 있습니다.

페이지 매김 사용 안 함

EEM은 명령 출력이 완료되었을 때 확인하는 장치 프롬프트를 찾습니다. 터미널 길이로 구성된 한 화면에 표시할 수 있는 것보다 많은 데이터를 출력하는 명령은 출력의 모든 페이지가 표시될 때까지 장치 프롬프트가 표시되지 않으므로 EEM 스크립트가 완료되고 결국 maxrun 타이머를 통해 종료되는 것을 방지할 수 있습니다. 큰 출력을 검토하는 EEM 스크립트 시작 시 용어 len을 구성합니다.

향후 유지보수에 대비한 설계 스크립트

EEM 스크립트를 디자인할 때는 작업 레이블 사이에 간격을 두어 나중에 EEM 스크립트 논리를 쉽게 업데이트할 수 있도록 합니다. 적절한 간격을 사용할 수 있는 경우(즉, action 0010 및 action 0020과 같은 두 명령문이 삽입 가능한 9개의 레이블 간격을 남기는 경우), 필요에 따라 작업 레이블의 번호를 다시 매기거나 다시 확인하지 않고 새 명령문을 추가할 수 있으며 작업이 계속해서 필요한 순서대로 실행되도록 할 수 있습니다.

EEM 스크립트 시작 부분에서 실행해야 하는 일반적인 명령이 있습니다. 여기에는 다음이 포함될 수 있습니다.

- 터미널 길이를 0으로 설정
- 활성화 모드로 들어갑니다.
- 명령 출력에 대한 자동 타임스탬프 활성화

이는 이 문서에 나와 있는 예에서 일반적인 패턴으로, 대부분의 스크립트는 이를 구성하기 위해 동일한 3개의 작업 문으로 시작합니다.

공통 EEM 로직 패턴

이 섹션에서는 EEM 스크립트에 사용되는 몇 가지 일반적인 논리 패턴과 구문 블록에 대해 설명합니다. 이 예제에서는 완전한 스크립트가 아니라 특정 기능을 사용하여 복잡한 EEM 스크립트를 만드는 방법을 보여 줍니다.

If/Else가 있는 브랜치 코드 경로

EEM 변수는 EEM 스크립트의 실행 흐름을 제어하는 데 사용할 수 있습니다. 이 EEM 스크립트를 고려해 보십시오.

```
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
```

이 스크립트는 매분마다 실행됩니다. CPU 사용률에 대한 SNMP OID 값을 검사한 다음 OID 값을 기준으로 세 가지 실행 경로 중 하나를 입력합니다. EEM 스크립트에서 복잡한 실행 흐름을 작성하기 위해 다른 모든 법적 EEM 변수에서 유사한 명령문을 사용할 수 있습니다.

Loop Over 문

실행 루프를 사용하여 EEM 스크립트를 크게 단축하고 추론을 쉽게 수행할 수 있습니다. 1분 동안 Te2/1/15에 대한 인터페이스 통계를 6번 풀링하여 활용률이 높은 소규모 기간을 확인하는 이 스크립트를 고려해 보십시오.

```
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"
```

EEM 루프 구문을 사용하면 이 스크립트를 크게 줄일 수 있습니다.

```
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
```

정규식을 통해 출력 추출(Regex)

EEM regexp 문을 사용하여 명령 출력에서 값을 추출하여 후속 명령에 사용할 수 있으며 EEM 스크립트 자체에서 동적 명령 생성을 활성화할 수 있습니다. show proc cpu의 출력에서 SNMP 엔진 PID를 추출하는 예는 이 코드 블록을 참조하십시오 | SNMP 엔진을 사용하여 syslog 메시지에 인쇄합니다. 이 추출된 값은 PID를 실행해야 하는 다른 명령에서도 사용할 수 있습니다.

```
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"
```

유용한 EEM 스크립트

MAC 주소에 대한 특정 MAC 주소 추적

이 예에서는 MAC 주소 b4e9.b0d3.6a41이 추적됩니다. 스크립트는 30초마다 검사하여 지정된 MAC 주소가 ARP 또는 MAC 테이블에서 학습되었는지 확인합니다. MAC가 표시되면 스크립트는 다음 작업을 수행합니다.

- syslog 메시지를 출력합니다(MAC 주소가 학습된 위치 또는 언제/얼마나 자주 학습되었는지 확인하려는 경우 유용합니다).

구현

```
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
```

SNMP OID를 통해 높은 CPU 모니터링

이 스크립트는 지난 5초 동안 CPU 사용 비율을 읽는 데 사용된 SNMP OID를 모니터링합니다. CPU가 80% 이상 사용 중인 경우 스크립트는 다음 작업을 수행합니다.

- show clock의 출력에서 타임스탬프를 만들고 이를 사용하여 고유한 파일 이름을 만듭니다
- 프로세스 및 소프트웨어 상태에 대한 출력이 이 파일에 기록됩니다
- epc(Embedded Packet Capture)는 컨트롤 플레인으로 향하는 10초의 트래픽을 캡처하여 파일에 쓰도록 구성됩니다.
- epc 캡처가 완료되면 EPC 컨피그레이션이 제거되고 스크립트가 종료됩니다.

구현

```
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"
```

PID와 레코드 스택 출력을 동적으로 일치

이 스크립트는 SNMP 입력 대기열이 가득 찼다는 syslog 메시지를 검색하고 다음 작업을 수행합니다.

- show proc cpu sort의 출력을 파일에 기록합니다.
- regex를 통해 SNMP 엔진 프로세스의 PID 추출
- 는 후속 명령에서 SNMP PID를 사용하여 PID에 대한 스택 데이터를 가져옵니다
- 더 이상 실행되지 않도록 구성에서 스크립트를 제거합니다.

구현

```
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"
```

스위치 업그레이드

이 스크립트는 install add file <file> activate commit 명령에서 반환되는 비표준 프롬프트에서 일치를 패턴화하고 프롬프트에 응답하도록 구성됩니다. 트리거 이벤트가 구성되지 않았으므로, UPGRADE를 실행하는 이벤트 관리자를 통해 업그레이드를 수행해야 할 때 사용자가 EEM 스크립트를 수동으로 트리거해야 합니다. maxrun 타이머는 install add 명령을 실행하는 데 상당한 시간이 소요되므로 기본값 20초가 아닌 300초로 설정됩니다.

구현

```
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"
```

IP SLA 추적 개체가 다운될 때 진단 데이터를 파일에 덤프

이 스크립트는 IP SLA 객체 11이 중단되고 다음 작업을 수행할 때 트리거됩니다.

- MAC 테이블, ARP 테이블, syslogs 및 라우팅 테이블 수집
- 플래시의 파일에 정보 쓰기: sla_track.txt

구현

```
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"
```

EEM에서 이메일 보내기

이 스크립트는 이벤트 syslog 패턴 문에 설명된 패턴이 표시될 때 트리거되며 다음 작업을 수행합니다.

- 내부 이메일 서버에서 이메일을 보냅니다(내부 이메일 서버가 디바이스에서 공개 인증을 허용한다고 가정).

구현

```
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
```

예약에서 포트 종료

이 스크립트는 매일 오후 6시에 포트 Te2/1/15를 종료합니다.

구현

```
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"
```

지정된 PPS(Packets Per Second) 속도에 도달하면 인터페이스 종료

이 스크립트는 매 초마다 TX 방향의 인터페이스 Te2/1/9에서 PPS 속도를 확인합니다. PPS 비율이 100을 초과하면 다음 작업을 수행합니다.

- 인터페이스에 대한show int 출력을 syslog에 기록합니다.
    - 인터페이스를 종료합니다.

구현

```
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 entry-type value
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"
```

관련 정보

- **Cisco EEM 모범 사례**
- **Cisco 기술 지원 및 다운로드**

이 번역에 관하여

Cisco는 전 세계 사용자에게 다양한 언어로 지원 콘텐츠를 제공하기 위해 기계 번역 기술과 수작업 번역을 병행하여 이 문서를 번역했습니다. 아무리 품질이 높은 기계 번역이라도 전문 번역가의 번역 결과물만큼 정확하지는 않습니다. Cisco Systems, Inc.는 이 같은 번역에 대해 어떠한 책임도 지지 않으며 항상 원본 영문 문서(링크 제공됨)를 참조할 것을 권장합니다.