Best practices en bruikbare scripts voor EEM begrijpen

Inhoud

Inleiding

Dit document beschrijft de best practices voor de configuratie van Embedded Event Manager (EEM) op Cisco IOS®-XE-apparaten en biedt voorbeelden van gebruikelijke syntax en nuttige scripts.

Voorwaarden

Vereisten

Dit document gaat ervan uit dat de lezer al bekend is met de functie Cisco IOS®/IOS-XE

Embedded Event Manager (EM). Als je nog niet bekend bent met deze functie, lees dan eerst het EEM Feature Overview.

Gebruikte componenten

De informatie in dit document is gebaseerd op de volgende software- en hardware-versies:

- Cisco Catalyst 9300, 9400 en 9500 switches waarop Cisco IOS-softwareversie 16.X of 17.X wordt uitgevoerd

Deze scripts worden niet ondersteund door Cisco TAC en worden op een as-is basis voor onderwijsdoeleinden geleverd.

De informatie in dit document is gebaseerd op de apparaten in een specifieke laboratoriumomgeving. Alle apparaten die in dit document worden beschreven, hadden een opgeschoonde (standaard)configuratie. Als uw netwerk live is, moet u zorgen dat u de potentiële impact van elke opdracht begrijpt.

Conventies

Raadpleeg de Cisco Technical Tips Conventions voor informatie over documentconventies.

Best practices

In dit deel worden enkele van de meest voorkomende problemen behandeld die bij het ontwerp en de implementatie van EEM-scripts zijn waargenomen. Voor aanvullende informatie over beste praktijken van de EEM, zie het EEM Best Practices document waarnaar wordt verwezen in de sectie Referenties.

Bevestig dat de juiste verificatie is uitgevoerd

Als uw apparaat AAA gebruikt, moet u ervoor zorgen dat de EEM-scripts die op het apparaat zijn geconfigureerd ofwel zijn geconfigureerd met een AAA-gebruiker die de opdrachten in het script kan uitvoeren, of dat de autorisatieomzeiling is geconfigureerd met de **opdrachtautorisatieomzeiling** in de scriptdefinitie.

Beperkingen voor EM-looptijd en -snelheidslimiet toevoegen

Standaard kunnen EEM-scripts maximaal 20 seconden draaien. Als u een script ontwerpt dat langer duurt om te starten, of moet wachten tussen het uitvoeren van de opdracht, specificeer dan een **maxRun-**waarde op de trigger van de applet-gebeurtenis om de standaard executie-timer te wijzigen.

Het is ook belangrijk om na te denken hoe vaak de gebeurtenis die uw EEM-script activeert, kan plaatsvinden. Als u een script activeert van een conditie die snel in een korte tijd optreedt (bijvoorbeeld syslog trigger voor MAC flaps), is het belangrijk om een snelheidslimietconditie op uw EEM script op te nemen om een excessief aantal executies parallel te voorkomen en uitputting van apparaatbronnen te voorkomen.

Vermijd out-of-order uitvoering

Zoals beschreven in de EEM-documentatie, wordt de volgorde van uitvoering voor actieverklaringen gecontroleerd door hun label (zo heeft de opdracht action 0001 cli **enable** een label van 0001). Deze labelwaarde is GEEN getal, maar eerder alfanumeriek. Acties worden gesorteerd in oplopende alfanumerieke toetsvolgorde, gebruik het **label** argument als de sorteertoets en ze worden in deze volgorde uitgevoerd. Dit kan leiden tot onverwachte volgorde van uitvoering, gebaseerd op hoe u structuur uw actie labels.

Neem dit voorbeeld:

```
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"
```

Aangezien 120 voor 13 in een alfanumerieke vergelijking is, loopt dit script niet in de volgorde die u verwacht. Om dit te voorkomen, is het nuttig om een systeem van vulling als dit te gebruiken:

```
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"
```

Vanwege de opvulling hier, de genummerde verklaringen evalueren in de verwachte volgorde. De toename van 10 tussen elk label maakt het mogelijk om later waar nodig extra verklaringen in het EEM-script in te voegen, zonder de noodzaak om alle volgende verklaringen te hernummeren.

Paginatie uitschakelen

EEM zoekt de apparatenherinnering om te bepalen wanneer de beveloutput volledig is. Opdrachten die de uitvoer meer gegevens dan kan worden weergegeven op één scherm (zoals geconfigureerd door uw eindlengte), kan voorkomen dat EEM scripts worden voltooid (en uiteindelijk gedood via de maxrun timer) als de apparaat prompt wordt niet weergegeven tot alle pagina's van de uitvoer worden bekeken. Configureer **term len 0** aan het begin van EEM-scripts die grote output onderzoeken.

Ontwerpscripts voor toekomstig onderhoud

Wanneer u een EEM-script ontwerpt, laat u ruimte tussen actielabels over om het in de toekomst gemakkelijker te maken om de EEM-scriptlogica bij te werken. Wanneer de nodige lacunes beschikbaar zijn (dat wil zeggen, twee verklaringen zoals **actie 0010** en **actie 0020** laten een gat van negen etiketten achter dat kan worden ingevoegd), kunnen zo nodig nieuwe verklaringen worden toegevoegd zonder de actielabels te hernummeren of opnieuw te controleren en ervoor te zorgen dat de acties in de verwachte volgorde blijven uitvoeren.

Er zijn gemeenschappelijke opdrachten die u moet uitvoeren aan het begin van uw EEM-scripts. Dit kan zijn:

- vastgestelde eindlengte aan 0
- activeert de activeringsmodus
- automatische tijdstempel voor opdrachtoutput inschakelen

Dit is een veelvoorkomend patroon in de voorbeelden die in dit document worden getoond, waar veel van de scripts beginnen met dezelfde 3 actieverklaringen om dit te configureren.

Gemeenschappelijke EEM Logic-patronen

Deze paragraaf behandelt enkele algemene logicapatronen en syntaxisblokken die in EEM-scripts worden gebruikt. De voorbeelden hier zijn niet complete scripts, maar eerder demonstraties van hoe specifieke functionaliteit kan worden gebruikt om complexe EEM scripts te maken.

Vestigingscodepaden met indien/anders

EEM variabelen kunnen worden gebruikt om de uitvoeringsstroom van EEM scripts te controleren. Bekijk dit EEM script:

```
event manager applet snmp_cpu authorization bypass
event timer watchdog time 60
action 0010 info type snmp oid 10.10.10.1.4.1.9.9.109.1.1.1.1.3.1 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
```

Dit script draait elke minuut. Onderzoek de waarde van SNMP OID voor het gebruik van CPU en voer vervolgens een van de drie verschillende executiepaden in op basis van de waarde van de OID. Gelijkaardige verklaringen kunnen op elke andere wettelijke EEM variabele worden gebruikt om complexe uitvoeringsstromen in EEM scripts op te bouwen.

Loop over verklaringen

Execution-loops kunnen worden gebruikt om EEM-scripts aanzienlijk te verkorten en ze gemakkelijker te begrijpen. Neem dit script, ontworpen om de interfacestatistieken voor Te2/1/15 6 keer over een periode van 1 minuut te trekken om te controleren op kleine periodes van hoog gebruik:

```
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"
```

Met **EEM loop constructs**, kan dit script aanzienlijk worden verkort:

```
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
```

Uitvoer via reguliere expressies (Regex)

De EEM regexp verklaring kan worden gebruikt om waarden uit opdrachtoutput te halen om in volgende opdrachten te worden gebruikt en om dynamische opdrachtcreatie binnen het EEM script zelf mogelijk te maken. Verwijs naar dit codeblok bij een voorbeeld om SNMP ENGINE PID uit de output van **show proc cpu** te halen **| i SNMP-engine** en druk deze af op een syslogbericht. Deze afgeleide waarde kan ook worden gebruikt in andere opdrachten waarvoor een PID moet worden uitgevoerd.

```
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"
```

Handige EEM-scripts

Track Specific MAC Address voor MAC Address Learn

In dit voorbeeld wordt het MAC-adres ***b4e9.b0d3.6a41*** bijgehouden. Het script controleert elke 30 seconden om te zien of het opgegeven MAC-adres is geleerd in de ARP- of MAC-tabellen. Als de MAC wordt gezien, voert het script deze acties:

- output een syslog bericht (dit is nuttig wanneer u wilt bevestigen waar een adres van MAC wordt geleerd, of wanneer/hoe vaak het wordt geleerd).

Implementatie

```
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
```

Hoge CPU's bewaken via SNMP OID

Dit script controleert een SNMP OID gebruikt om het CPU bezig percentage in de laatste 5 seconden te lezen. Wanneer de CPU meer dan 80% bezig is, voert het script de volgende handelingen uit:

- creëert een tijdstempel van de output van show klok, en gebruikt dit om een unieke bestandsnaam te maken
- de uitgangen over proces en softwarestatus worden vervolgens naar dit bestand geschreven
- Een Embedded Packet Capture (EPC) is geconfigureerd om 10 seconden verkeer op te nemen dat is bestemd voor het besturingsplane en schrijft naar een bestand.
- zodra EPC-opname is voltooid, wordt de EPC-configuratie verwijderd en gaat het script terug.

Implementatie

```
event manager applet high-cpu authorization bypass
event snmp oid 10.10.10.1.4.1.9.9.109.1.1.1.1.3.1 get-type next entry-op gt entry-val 80 poll-
interval 1 ratelimit 300 maxrun 180
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-$time.txt"
action 0150 cli command "show platform software fed switch active punt cause summary | append
flash:tac-cpu-$time.txt"
action 0160 cli command "show platform software fed switch active cpu-interface | append
flash:tac-cpu-$time.txt"
action 0170 cli command "show platform software fed switch active punt cpuq all | append
flash:tac-cpu-$time.txt"
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-$time.pcap"
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"
```

Dynamisch een PID- en een opnamestapeluitvoer aanpassen

Dit script zoekt naar een syslogbericht dat de SNMP-invoerwachtrij vol is en voert de volgende acties uit:

- legt de uitvoer van de **show project cpu soort** op een bestand
- extraheert de PID van het SNMP ENGINE proces via regex
- gebruikt de SNMP PID in volgende opdrachten om de stackgegevens voor de PID te verkrijgen

- verwijdert het script uit de configuratie zodat er geen executies meer van het voorkomen

Implementatie

```
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"
```

Upgrade een Switch

Dit script is ingesteld om de niet-standaard prompt die door de **install add bestand <bestand> activeert commit** opdracht te koppelen en te reageren op de aanwijzingen. Geen trigger-gebeurtenis is ingesteld, dus het EEM-script moet handmatig worden geactiveerd door een gebruiker wanneer de upgrade moet plaatsvinden via **event manager run UPGRADE**. De maximale timer wordt ingesteld op 300 seconden in plaats van de standaardwaarde van 20 seconden, aangezien de opdracht **Add installeren** een aanzienlijke tijd in beslag neemt.

Implementatie

```
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 "y\/n" action 0030 cli command "y"
pattern "y\/n" action 0040 syslog msg "Reloading device to upgrade code" action 0050 cli command
"y"
```

Diagnostische gegevens in een bestand dumpen wanneer een door IP SLA gevolgd object naar beneden gaat

Dit script wordt geactiveerd wanneer IP SLA object 11 neergaat en deze handelingen uitvoert:

- Verzamel MAC-tabel, ARP-tabel, syslogs en Routing-tabel
- Schrijf informatie naar een bestand op flash: genaamd sla_track.txt

Implementatie

```
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"
```

Verzend een e-mail van EEM

Dit script wordt getriggerd wanneer het patroon beschreven in het gebeurtenissyslog patroonstatement wordt gezien, en voert deze acties:

- stuurt een e-mail vanaf een interne e-mailserver (hierbij wordt ervan uitgegaan dat de interne e-mailserver open verificatie vanaf het apparaat mogelijk maakt).

Implementatie

```
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 - Syslog seen on $_info_routername" body “BODY OF YOUR EMAIL GOES HERE”
```

Een poort op een schema afsluiten

Dit script sluit poort Te2/1/15 elke dag af om 18:00 uur.

Implementatie

```
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"
```

Sluit een interface als een bepaalde pakketten per seconde (PPS) tarief wordt bereikt

Dit script controleert elke seconde de PPS-snelheid op interface Te2/1/9 in de TX richting. Als het PPS-percentage hoger is dan 100, wordt de volgende actie ondernomen:

- logt de **show int.** uitvoer voor de interface aan syslog in
- sluit de interface

Implementatie

```
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"
```

Referenties

- Beste praktijken van Cisco EM

Over deze vertaling

Cisco heeft dit document vertaald via een combinatie van machine- en menselijke technologie om onze gebruikers wereldwijd ondersteuningscontent te bieden in hun eigen taal. Houd er rekening mee dat zelfs de beste machinevertaling niet net zo nauwkeurig is als die van een professionele vertaler. Cisco Systems, Inc. is niet aansprakelijk voor de nauwkeurigheid van deze vertalingen en raadt aan altijd het oorspronkelijke Engelstalige document (link) te raadplegen.