Comprendere le procedure ottimali e gli script utili per EEM

Sommario

Introduzione

In questo documento vengono descritte le best practice per la configurazione degli script di Embedded Event Manager (EEM) sui dispositivi Cisco IOS®-XE e vengono forniti esempi di sintassi comune e script utili.

Prerequisiti

Requisiti

in questo documento si presume che il lettore abbia già familiarità con la funzionalità Cisco IOS®/IOS-XE Embedded Event Manager (EEM). Se non si ha familiarità con questa funzione,

leggere prima la panoramica della funzione EEM.

Componenti usati

Le informazioni fornite in questo documento si basano sulle seguenti versioni software e hardware:

- Switch Cisco Catalyst 9300, 9400 e 9500 con software Cisco IOS versione 16.X o 17.X

Questi script non sono supportati da Cisco TAC e vengono forniti così come sono per scopi educativi.

Le informazioni discusse in questo documento fanno riferimento a dispositivi usati in uno specifico ambiente di emulazione. Su tutti i dispositivi menzionati nel documento la configurazione è stata ripristinata ai valori predefiniti. Se la rete è operativa, valutare attentamente eventuali conseguenze derivanti dall'uso dei comandi.

Convenzioni

Per ulteriori informazioni sulle convenzioni usate, consultare il documento Cisco sulle convenzioni nei suggerimenti tecnici.

Procedure ottimali

In questa sezione vengono illustrati alcuni dei problemi più comuni osservati nella progettazione e nell'implementazione di script EEM. Per ulteriori informazioni sulle procedure consigliate di Manutenzione di Internet Explorer, vedere il documento sulle procedure consigliate di Manutenzione di Internet Explorer a cui si fa riferimento nella sezione Riferimenti.

Conferma l'autenticazione appropriata

Se il dispositivo usa il server AAA, accertarsi che gli script EEM configurati sul dispositivo siano configurati con un utente AAA in grado di eseguire i comandi nello script o che il bypass autorizzazione sia configurato con il **bypass autorizzazione** comando nella definizione dello script.

Aggiungi vincoli per runtime EEM e limite di velocità

Per impostazione predefinita, gli script EEM possono essere eseguiti per un massimo di 20 secondi. Se si progetta uno script che impiega più tempo per l'esecuzione o che deve attendere tra l'esecuzione del comando, specificare un valore **maxrun** sul trigger di evento applet per modificare il timer di esecuzione predefinito.

È inoltre importante considerare la frequenza di esecuzione dell'evento che attiva lo script EEM. Se si attiva uno script da una condizione che si verifica rapidamente in un breve periodo di tempo (ad esempio, il trigger syslog per i flap MAC), è importante includere una condizione di limite di velocità nello script EEM per evitare un numero eccessivo di esecuzioni in parallelo e l'esaurimento delle risorse del dispositivo.

Evitare l'esecuzione non ordinata

Come descritto nella documentazione EEM, l'ordine di esecuzione delle istruzioni di azione è controllato dalla relativa etichetta (ad esempio, il comando **enable** dell'azione 0001 della cli ha un'etichetta 0001). Il valore dell'etichetta NON è un numero, ma piuttosto un valore alfanumerico. Le azioni vengono ordinate in sequenza di tasti alfanumerici ascendenti, utilizzando l'argomento **etichetta** come chiave di ordinamento e vengono eseguite in questa sequenza. Ciò può determinare un ordine di esecuzione imprevisto, in base alla struttura delle etichette delle azioni.

Considerate questo esempio:

```
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"
```

Poiché 120 è precedente a 13 in un confronto alfanumerico, questo script non viene eseguito nell'ordine previsto. Per evitare ciò, è utile utilizzare un sistema di spaziatura interna simile al seguente:

```
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"
```

A causa della spaziatura interna qui, le istruzioni numerate vengono valutate nell'ordine previsto. L'incremento di 10 tra ciascuna etichetta consente di inserire istruzioni aggiuntive nello script EEM in un secondo momento, se necessario, senza la necessità di rinumerare tutte le istruzioni successive.

Disabilita paginazione

EEM cerca il prompt del dispositivo per determinare quando l'output del comando è completo. I comandi che generano più dati di quanti possano essere visualizzati su una schermata (come configurato dalla lunghezza del terminale) possono impedire il completamento degli script EEM (e alla fine vengono terminati con il timer maxrun) poiché il prompt del dispositivo non viene visualizzato finché non vengono visualizzate tutte le pagine dell'output. Configurare la **len di termini 0** all'inizio degli script EEM che esaminano gli output di grandi dimensioni.

Script di progettazione per la manutenzione futura

Quando si progetta uno script EEM, lasciare spazi tra le etichette delle azioni per semplificare l'aggiornamento futuro della logica dello script EEM. Quando sono disponibili spazi vuoti appropriati, ovvero due istruzioni come **azione 0010** e **azione 0020** lasciano un intervallo di 9 etichette che possono essere inserite, è possibile aggiungere nuove istruzioni come richiesto senza rinumerare o ricontrollare le etichette delle azioni e garantire che le azioni continuino a essere eseguite nell'ordine previsto.

Ènecessario eseguire alcuni comandi comuni all'inizio degli script EEM. Ciò può includere:

- imposta la lunghezza del terminale su 0
- accedere alla modalità di abilitazione
- abilita indicatore orario automatico per l'output del comando

Si tratta di uno schema comune negli esempi riportati in questo documento, in cui molti script

iniziano con le stesse 3 istruzioni action per configurare questa condizione.

Modelli logici EEM comuni

In questa sezione vengono illustrati alcuni modelli logici e blocchi di sintassi comuni utilizzati negli script EEM. Gli esempi riportati non sono script completi, ma dimostrazioni di come sia possibile utilizzare funzionalità specifiche per creare script EEM complessi.

Percorsi codice filiale con if/else

Le variabili EEM possono essere utilizzate per controllare il flusso di esecuzione degli script EEM. Si consideri il seguente script EEM:

```
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
```

Questo script viene eseguito ogni minuto. Esaminare il valore dell'OID SNMP per l'utilizzo della CPU e quindi immettere uno dei tre percorsi di esecuzione diversi in base al valore dell'OID. Istruzioni simili possono essere utilizzate in qualsiasi altra variabile EEM legale per creare flussi di esecuzione complessi negli script EEM.

Istruzioni Loop Over

I loop di esecuzione possono essere utilizzati per abbreviare in modo significativo gli script EEM e semplificarne la comprensione. Prendiamo in considerazione questo script, progettato per estrarre le statistiche dell'interfaccia per Te2/1/15 6 volte in un periodo di 1 minuto per controllare i piccoli periodi di elevato utilizzo:

```
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"
```

Con i **costrutti loop EEM**, questo script può essere significativamente abbreviato:

```
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
```

Estrai output tramite espressioni regolari (Regex)

L'istruzione regexp EEM può essere utilizzata per estrarre valori dall'output del comando da utilizzare nei comandi successivi e abilitare la creazione dinamica dei comandi all'interno dello script EEM stesso. Fare riferimento a questo blocco di codice per un esempio per estrarre il PID del motore SNMP dall'output del comando **show proc cpu | i Motore SNMP** e stamparlo su un messaggio syslog. Questo valore estratto può essere utilizzato anche in altri comandi che richiedono l'esecuzione di un PID.

```
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"
```

Script EEM utili

Tieni traccia dell'indirizzo MAC specifico per l'indirizzo MAC

Nell'esempio, viene tracciato l'indirizzo MAC ***b4e9.b0d3.6a41***. Lo script controlla ogni 30 secondi per verificare se l'indirizzo MAC specificato è stato appreso nelle tabelle ARP o MAC. Se l'indirizzo MAC viene visualizzato, lo script esegue le azioni seguenti:

- genera un messaggio syslog (utile quando si desidera confermare dove viene appreso un indirizzo MAC o quando/quanto spesso viene appreso).

Implementazione

```
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
```

Monitoraggio di CPU elevate tramite SNMP OID

Questo script controlla un OID SNMP utilizzato per leggere la percentuale di utilizzo della CPU negli ultimi 5 secondi. Quando la CPU è occupata per oltre l'80%, lo script esegue le azioni seguenti:

- crea un timestamp dall'output di show clock e lo utilizza per creare un nome file univoco
- gli output relativi allo stato del processo e del software vengono quindi scritti in questo file
- un EPC (Embedded Packet Capture) è configurato per acquisire 10 secondi di traffico destinato al control-plane e lo scrive in un file.
- al termine dell'acquisizione EPC, la configurazione EPC viene rimossa e lo script viene chiuso.

Implementazione

```
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"
```

Corrispondenza dinamica di un PID e dell'output di uno stack di record

Questo script cerca un messaggio syslog per segnalare che la coda di input SNMP è piena ed esegue le azioni seguenti:

- registra l'output dell'**ordinamento show proc cpu** in un file

- estrae il PID del processo SNMP ENGINE tramite regex
- utilizza il PID SNMP nei comandi successivi per ottenere i dati dello stack per il PID
- rimuove lo script dalla configurazione in modo che non venga più eseguito

Implementazione

```
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"
```

Aggiornamento di uno switch

Questo script è configurato in modo da corrispondere al modello nel prompt non standard restituito dal comando **install add file <file> activate commit** e rispondere ai prompt. Poiché non è configurato alcun evento trigger, lo script EEM deve essere attivato manualmente da un utente quando è necessario eseguire l'aggiornamento tramite **gestione eventi per eseguire UPGRADE**. Il timer maxrun viene impostato per 300 secondi anziché per il valore predefinito di 20 secondi, in quanto l'esecuzione del comando **install add** richiede molto tempo.

Implementazione

```
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"
```

Dump dei dati di diagnostica in un file quando un oggetto registrato con SLA IP diventa inattivo

Questo script viene attivato quando l'oggetto 11 del contratto di servizio IP diventa inattivo e esegue le azioni seguenti:

- Raccogli tabella MAC, tabella ARP, syslog e tabella di routing
- Scrivere le informazioni in un file su flash: denominato sla_track.txt

Implementazione

```
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"
```

Invia un messaggio di posta elettronica da EEM

Questo script viene attivato quando viene visualizzato il modello descritto nell'istruzione event syslog pattern ed esegue le azioni seguenti:

- invia un messaggio di posta elettronica da un server di posta elettronica interno (si presume che il server di posta elettronica interno consenta l'autenticazione aperta dal dispositivo).

Implementazione

```
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”
```

Arresto di una porta su una pianificazione

Questo script chiude la porta Te2/1/15 ogni giorno alle 18.

Implementazione

```
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"
```

Arresta un'interfaccia se viene raggiunta una determinata velocità di pacchetti al secondo (PPS)

Questo script controlla la velocità PPS sull'interfaccia Te2/1/9 in direzione TX ogni secondo. Se la velocità di PPS è superiore a 100, vengono eseguite le azioni seguenti:

- registra l'output **show int** per l'interfaccia in syslog
- chiude l'interfaccia

Implementazione

```
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"
```

Riferimenti

- Procedure ottimali EEM Cisco

Informazioni su questa traduzione

Cisco ha tradotto questo documento utilizzando una combinazione di tecnologie automatiche e umane per offrire ai nostri utenti in tutto il mondo contenuti di supporto nella propria lingua. Si noti che anche la migliore traduzione automatica non sarà mai accurata come quella fornita da un traduttore professionista. Cisco Systems, Inc. non si assume alcuna responsabilità per l’accuratezza di queste traduzioni e consiglia di consultare sempre il documento originale in inglese (disponibile al link fornito).