# Best Practices und nützliche Skripte für EEM

## Inhalt



## Einleitung

Dieses Dokument beschreibt die Best Practices für die EEM-Skriptkonfiguration (Embedded Event Manager) auf Cisco IOS®-XE-Geräten und enthält Beispiele für gängige Syntax und nützliche Skripts.

## Voraussetzungen

### Anforderungen

In diesem Dokument wird davon ausgegangen, dass der Leser bereits mit der Funktion des Cisco IOS®/IOS-XE Embedded Event Manager (EEM) vertraut ist. Wenn Sie diese Funktion noch nicht kennen, lesen Sie zunächst die EEM-Funktionsübersicht.

### Verwendete Komponenten

Die Informationen in diesem Dokument basierend auf folgenden Software- und Hardware-Versionen:

- Cisco Catalyst Switches der Serien 9300, 9400 und 9500 mit Cisco IOS Software Version 16.X oder 17.X

**Diese Skripte werden vom Cisco TAC nicht unterstützt und werden zu Informationszwecken wie besehen bereitgestellt.**

Die Informationen in diesem Dokument beziehen sich auf Geräte in einer speziell eingerichteten Testumgebung. Alle Geräte, die in diesem Dokument benutzt wurden, begannen mit einer gelöschten (Nichterfüllungs) Konfiguration. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die möglichen Auswirkungen aller Befehle verstehen.

### Konventionen

Informationen zu Dokumentkonventionen finden Sie unter Cisco Technical Tips Conventions (Technische Tipps von Cisco zu Konventionen).

## Best Practices

In diesem Abschnitt werden einige der häufigsten Probleme beim Design und bei der Implementierung von EEM-Skripts behandelt. Weitere Informationen zu EEM Best Practices finden Sie im EEM Best Practices-Dokument, auf das im Abschnitt Referenzen verwiesen wird.

### Bestätigen Sie die korrekte Authentifizierung.

Wenn Ihr Gerät AAA verwendet, müssen Sie sicherstellen, dass die auf dem Gerät konfigurierten EEM-Skripts entweder mit einem AAA-Benutzer konfiguriert sind, der die Befehle im Skript ausführen kann, oder dass die Autorisierungs-Umgehung mit der Befehl-**Autorisierungs-Umgehung** in der Skriptdefinition konfiguriert ist.

### Hinzufügen von Einschränkungen für EEM-Laufzeit und Ratenlimit

Standardmäßig können EEM-Skripts für maximal 20 Sekunden ausgeführt werden. Wenn Sie ein Skript entwerfen, dessen Ausführung länger dauert oder das zwischen der Befehlsausführung warten muss, geben Sie einen **maxrun-**Wert auf dem Ereignisauslöser des Applets an, um den Standardausführungszeitgeber zu ändern.

Es ist auch wichtig zu berücksichtigen, wie oft das Ereignis, das das EEM-Skript auslöst, ausgeführt werden kann. Wenn Sie ein Skript aufgrund einer Bedingung auslösen, die in kurzer Zeit sehr schnell auftritt (z. B. Syslog-Auslöser für MAC-Flaps), muss das EEM-Skript eine Durchsatzbegrenzungsbedingung enthalten, um zu viele parallele Ausführungen zu verhindern und eine Erschöpfung der Geräteressourcen zu verhindern.

### Vermeiden von ungeordneter Ausführung

Wie in der EEM-Dokumentation beschrieben, wird die Ausführungsreihenfolge für Action-

Anweisungen durch ihr Label gesteuert (Beispiel: action 0001 cli command **enable** hat das Label 0001). Dieser Labelwert ist KEINE Zahl, sondern eher alphanumerisch. Aktionen werden in aufsteigender alphanumerischer Schlüsselsequenz sortiert, das **Label-**Argument als Sortierschlüssel verwendet und in dieser Sequenz ausgeführt. Dies kann zu einer unerwarteten Reihenfolge der Ausführung führen, je nachdem, wie Sie Ihre Aktionsbezeichnungen strukturieren.

Betrachten Sie dieses Beispiel:

```
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"
```

Da 120 in einem alphanumerischen Vergleich vor 13 liegt, läuft dieses Skript nicht in der erwarteten Reihenfolge. Um dies zu vermeiden, ist es sinnvoll, ein System der Polsterung wie folgt zu verwenden:

```
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"
```

Durch die Auffüllung hier werden die nummerierten Aussagen in der erwarteten Reihenfolge ausgewertet. Durch das Inkrement von 10 zwischen jedem Label können bei Bedarf später zusätzliche Anweisungen in das EEM-Skript eingefügt werden, ohne dass alle nachfolgenden Anweisungen neu nummeriert werden müssen.

## Paginierung deaktivieren

EEM sucht nach der Geräteaufforderung, um zu bestimmen, wann die Befehlsausgabe abgeschlossen ist. Befehle, die mehr Daten ausgeben, als auf einem Bildschirm angezeigt werden können (entsprechend der Terminallänge), können den Abschluss von EEM-Skripts verhindern (und schließlich über den maxrun-Timer beendet werden), da die Geräteaufforderung erst angezeigt wird, wenn alle Seiten der Ausgabe angezeigt werden. Konfigurieren Sie **den Ausdruck len 0** zu Beginn von EEM-Skripts, die große Ausgaben untersuchen.

## Design-Skripte für künftige Wartungsarbeiten

Lassen Sie beim Entwerfen eines EEM-Skripts Lücken zwischen Aktionsbezeichnungen, damit die EEM-Skriptlogik in Zukunft leichter aktualisiert werden kann. Wenn entsprechende Lücken vorhanden sind (d. h. zwei Anweisungen wie **Aktion 0010** und **Aktion 0020** lassen eine Lücke von 9 Beschriftungen, die eingefügt werden können), können neue Anweisungen nach Bedarf hinzugefügt werden, ohne die Beschriftungen neu zu nummerieren oder erneut zu überprüfen, und stellen sicher, dass die Aktionen weiterhin in der erwarteten Reihenfolge ausgeführt werden.

Es gibt einige gängige Befehle, die Sie zu Beginn Ihrer EEM-Skripts ausführen müssen. Dies kann Folgendes umfassen:

- Klemmenlänge auf 0 einstellen
- Aktivierungsmodus eingeben
- Automatische Zeitstempel für Befehlsausgabe aktivieren

Dies ist ein gängiges Muster in den Beispielen in diesem Dokument, in denen viele der Skripte mit

den gleichen drei action-Anweisungen beginnen, um dies zu konfigurieren.

# Allgemeine EEM-Logikmuster

Dieser Abschnitt behandelt einige gängige Logikmuster und Syntaxblöcke, die in EEM-Skripts verwendet werden. Bei den Beispielen handelt es sich nicht um vollständige Skripte, sondern vielmehr um Demonstrationen, wie spezifische Funktionen zum Erstellen komplexer EEM-Skripte verwendet werden können.

## Verzweigungscodepfade mit if/else

EEM-Variablen können verwendet werden, um den Ausführungsfluss von EEM-Skripts zu steuern. Betrachten Sie dieses EEM-Skript:

```
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
```

Dieses Skript läuft jede Minute. Überprüfen Sie den Wert der SNMP-OID auf die CPU-Nutzung, und geben Sie dann einen von drei verschiedenen Ausführungspfaden ein, die auf dem Wert der OID basieren. Ähnliche Anweisungen können für jede andere zulässige EEM-Variable verwendet werden, um komplexe Ausführungsflüsse in EEM-Skripts zu erstellen.

## Loop Over-Anweisungen

Mithilfe von Execution Loops können EEM-Skripte deutlich verkürzt und ihre Argumentation vereinfacht werden. Betrachten Sie dieses Skript, das entwickelt wurde, um die Schnittstellenstatistiken für Te2/1/15 6 Mal innerhalb eines Zeitraums von 1 Minute zu laden, um zu überprüfen, ob kleine Zeiträume mit hoher Auslastung vorliegen:

```
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"
```

Mit **EEM-Schleifenkonstrukten** kann dieses Skript erheblich verkürzt werden:

```
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
```

### Ausgabe über reguläre Ausdrücke extrahieren (Regex)

Die EEM-regexp-Anweisung kann verwendet werden, um Werte aus der Befehlsausgabe zu extrahieren, die in nachfolgenden Befehlen verwendet werden sollen, und um die dynamische Befehlserstellung innerhalb des EEM-Skripts selbst zu ermöglichen. In diesem Codeblock finden Sie ein Beispiel zum Extrahieren der SNMP ENGINE PID aus der Ausgabe von **show proc cpu | die SNMP-Engine** und druckt sie als Syslog-Meldung aus. Dieser extrahierte Wert kann auch in anderen Befehlen verwendet werden, für die eine PID ausgeführt werden muss.

```
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"
```

## Nützliche EEM-Skripts

### Spezifische MAC-Adresse verfolgen MAC-Adresse erfahren

In diesem Beispiel wird die MAC-Adresse ***b4e9.b0d3.6a41*** nachverfolgt. Das Skript überprüft alle 30 Sekunden, ob die angegebene MAC-Adresse in den ARP- oder MAC-Tabellen erfasst wurde. Wenn die MAC-Adresse angezeigt wird, führt das Skript folgende Aktionen aus:

- gibt eine Syslog-Meldung aus (dies ist nützlich, wenn Sie bestätigen möchten, wo oder wann eine MAC-Adresse erfasst wird).

#### Implementierung

```
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
```

## Hohe CPU über SNMP OID überwachen

Dieses Skript überwacht eine SNMP-OID, die in den letzten 5 Sekunden zum Lesen des prozentualen CPU-Auslastungsgrads verwendet wurde. Wenn die CPU zu mehr als 80 % ausgelastet ist, führt das Skript folgende Aktionen aus:

- erstellt einen Zeitstempel aus der Ausgabe der Uhr und verwendet diesen, um einen eindeutigen Dateinamen zu erstellen
- Ausgaben über Prozess- und Softwarestatus werden dann in diese Datei geschrieben.
- Eine Embedded Packet Capture (EPC) ist so konfiguriert, dass sie 10 Sekunden Datenverkehr erfasst, der für die Steuerungsebene bestimmt ist, und diese in eine Datei schreibt.
- Nach Abschluss der EPC-Erfassung wird die EPC-Konfiguration entfernt, und das Skript wird beendet.

### Implementierung

```
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"
```

## Dynamische Zuordnung von PID und Datensatzstapel-Ausgabe

Dieses Skript sucht nach einer Syslog-Meldung, dass die SNMP-Eingabewarteschlange voll ist,

und führt die folgenden Aktionen aus:

- zeichnet die Ausgabe von **show proc cpu sort** in eine Datei auf
- extrahiert die PID des SNMP ENGINE-Prozesses über regex
- verwendet die SNMP-PID in nachfolgenden Befehlen, um die Stapeldaten für die PID abzurufen.
- entfernt das Skript aus der Konfiguration, sodass es nicht mehr ausgeführt wird

### Implementierung

```
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"
```

## Switch-Upgrade

Dieses Skript ist so konfiguriert, dass die Musterübereinstimmung an der vom Befehl **install add file <file> activate commit** zurückgegebenen nicht standardmäßigen Eingabeaufforderung erfolgt und auf die Eingabeaufforderungen reagiert. Es ist kein Triggerereignis konfiguriert. Daher muss das EEM-Skript manuell von einem Benutzer ausgelöst werden, wenn das Upgrade über den **Ereignis-Manager** erfolgen muss, und **führen Sie UPGRADE aus**. Der Timer maxrun wird für 300 Sekunden statt des Standardwerts von 20 Sekunden festgelegt, da die Ausführung des Befehls **install add** sehr viel Zeit in Anspruch nimmt.

### Implementierung

```
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"
```

## Diagnosedaten in eine Datei speichern, wenn ein IP SLA-Objekt ausfällt

Dieses Skript wird ausgelöst, wenn das IP SLA-Objekt 11 ausfällt und folgende Aktionen ausführt:

- Erfassen der MAC-Tabelle, ARP-Tabelle, Syslogs und Routing-Tabelle
- Informationen in eine Datei im Flash schreiben: sla_track.txt aufgerufen

### Implementierung

```
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"
```

## E-Mail von EEM senden

Dieses Skript wird ausgelöst, wenn das in der Syslog-Pattern-Anweisung des Ereignisses beschriebene Muster erkannt wird, und ergreift die folgenden Maßnahmen:

- sendet eine E-Mail von einem internen E-Mail-Server (dies setzt voraus, dass der interne E-Mail-Server eine offene Authentifizierung vom Gerät ermöglicht).

### Implementierung

```
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”
```

## Abschaltung eines Ports nach einem Zeitplan

Dieses Skript fährt den Port Te2/1/15 täglich um 18 Uhr herunter.

### Implementierung

```
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"
```

## Herunterfahren einer Schnittstelle, wenn eine bestimmte Paketrate pro Sekunde (PPS) erreicht wird

Dieses Skript überprüft die PPS-Rate an der Schnittstelle Te2/1/9 in TX-Richtung jede Sekunde. Wenn die PPS-Rate 100 übersteigt, werden folgende Schritte ausgeführt:

- protokolliert die Ausgabe von **show int** für die Schnittstelle in syslog
- fährt die Schnittstelle herunter

### Implementierung

```
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"
```

# Referenzen

- Cisco EEM - Best Practices

Informationen zu dieser Übersetzung

Cisco hat dieses Dokument maschinell übersetzen und von einem menschlichen Übersetzer editieren und korrigieren lassen, um unseren Benutzern auf der ganzen Welt Support-Inhalte in ihrer eigenen Sprache zu bieten. Bitte beachten Sie, dass selbst die beste maschinelle Übersetzung nicht so genau ist wie eine von einem professionellen Übersetzer angefertigte. Cisco Systems, Inc. übernimmt keine Haftung für die Richtigkeit dieser Übersetzungen und empfiehlt, immer das englische Originaldokument (siehe bereitgestellter Link) heranzuziehen.