أستكشاف أخطاء الوحدة النمطية WANon
وإصلاحها والتحقق من صحتها في CGR 1000

المحتويات

المقدمة

يوضح هذا المستند كيفية أستكشاف أخطاء لغة أمر الأداة (TCL) وإصلاحها باسم tm_wanmon.tcl المدمج في صور IOS لموجه الشبكة المتصلة (CGR) من السلسلة Cisco IOS1000. يعمل هذا البرنامج النصي، عند تمكينه، كبرنامج مراقبة شبكة المنطقة الواسعة (WAN) لواجهات WAN ويقوم بتنفيذ إجراءات اقتحامية بشكل متزايد للتمكن من التعافي من فشل إرتباط WAN أو النفق.

عندما يتم تمكين WANmon على الموجه، يتم إستخراج البرنامج النصي في الدليل:tmpsys:/im_policy/ ويمكن عرض البرنامج النصي باستخدام هذا الأمر؛ **Router#more tmpsys:eem_policy/tm_wanmon.tcl**.

المتطلبات الأساسية

المتطلبات

cisco يوصي أن يمكن أنت TCL نص كما هو موضح في هذا وثيقة؛ http://www.cisco.com/c/en/us/td/docs/routers/connectedgrid/cgr1000/ios/software/15_4_1_cg/WANMon.html.

الحد الأدنى للبرامج: الإصدار CG(1)15.4 من Cisco IOS.

المكونات المستخدمة

تستند المعلومات الواردة في هذا المستند إلى CGR1120/CGR1240.

تم إنشاء المعلومات الواردة في هذا المستند من الأجهزة الموجودة في بيئة معملية خاصة. بدأت جميع الأجهزة المُستخدمة في هذا المستند بتكوين ممسوح (افتراضي). إذا كانت شبكتك مباشرة، فتأكد من فهمك للتأثير المحتمل لأي أمر.

أستكشاف أخطاء MethodOloy وإصلاحها

عند تكوين WANmon وتشغيله على CGR، يمكن تمكين تصحيح الأخطاء للبرنامج النصي عند إضافة هذا التكوين إلى تكوين الجهاز:

```
Router(config)#event manager environment wanmon_debug all
```

التحقق من البرنامج النصي

```
Router(config)#event manager policy tm_wanmon.tcl authorization bypass
Router(config)#event manager environment wanmon_if_list1 {GigabitEthernet2/1 {recovery {5 25}
{ {{{90 180} {600 100
Router(config)#event manager environment wanmon_debug all

Aug 8 21:24:14.555: %HA_EM-6-LOG: tm_wanmon.tcl: Command: enable     <----- WANMON is orienting*
itself to privileged exec mode by sending the 'enable' command to IOS
:Aug 8 21:24:14.775: %HA_EM-6-LOG: tm_wanmon.tcl: Output*
Aug 8 21:24:14.775: %HA_EM-6-LOG: tm_wanmon.tcl: CGR1000#     <----- Output from IOS tells*
WANMON that it is now at the privileged exec prompt
Aug 8 21:24:14.789: %HA_EM-6-LOG: tm_wanmon.tcl: context_retrieve succeeded     <----- The*
router was able to successfully retrieve WANMON polling data
Aug 8 21:24:14.789: %HA_EM-6-LOG: tm_wanmon.tcl: last_run: 1249760980     <----- The event*
identifier for the last time WANMON polled its enabled interfaces
Aug 8 21:24:14.789: %HA_EM-6-LOG: tm_wanmon.tcl: exec_count: 5     <----- The number of times*
WANMON has polled since it last started. The last operation (1249760980) was the 5th process in
a row
Aug 8 21:24:14.791: %HA_EM-6-LOG: tm_wanmon.tcl: GigabitEthernet2/1,link_events: {1 {}*
1249760380 {Sat Aug 08 19:39:40 UTC 2009}}     <----- Enabled interfaces and time stamp of first
execution
Aug 8 21:24:14.791: %HA_EM-6-LOG: tm_wanmon.tcl: last_polled_run: 1249760980     <-----*
last_run and last_polled_run are equal in this case but if recovery levels are active the last
polled run may be earlier than the last_run
Aug 8 21:24:14.797: %HA_EM-6-LOG: tm_wanmon.tcl: Starting: Received event polltimer     <-----*
A new polling process has started with a new polltimer
{} :Aug 8 21:24:14.799: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_name Value*
Aug 8 21:24:14.799: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_pub_sec Value: 1249766654     <---*
-- The new event identifier
Aug 8 21:24:14.799: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_type Value: 21*
Aug 8 21:24:14.801: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_remain Value: 600.000     <-----*
The polltimer has been reset and there are 600 seconds until the next polling event
Aug 8 21:24:14.801: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_severity Value: severity-normal   *
<----- This message indicates that this is a regular poll and not caused by one of the alert
levels being triggered from a previous polling event
Aug 8 21:24:14.803: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_id Value: 3*
Aug 8 21:24:14.803: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_type_string Value: timer watchdog*
Aug 8 21:24:14.803: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_time_sec Value: 3458755454*
Aug 8 21:24:14.805: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_type Value: watchdog*
Aug 8 21:24:14.805: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_remain_msec Value: 0*
Aug 8 21:24:14.805: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_remain_sec Value: 600*
Aug 8 21:24:14.805: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_time_msec Value: 46*
Aug 8 21:24:14.807: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_pub_time Value: 1249766654.045*
Aug 8 21:24:14.807: %HA_EM-6-LOG: tm_wanmon.tcl: Name: job_id Value: 6*
Aug 8 21:24:14.807: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_pub_msec Value: 45*
Aug 8 21:24:14.807: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_time Value: 3458755454.046*
Aug 8 21:24:14.807: %HA_EM-6-LOG: tm_wanmon.tcl: Exec count: 6     <----- The exec_count has*
incremented by 1 from the previous execution
Aug 8 21:24:14.811: %HA_EM-6-LOG: tm_wanmon.tcl: All interface vars wanmon_if_list1     <-----*
If multiple lists are present, they will all be included here separated by spaces (i.e.
(wanmon_if_list1 wanmon_if_list2 wanmon_if_list3
Aug 8 21:24:14.813: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_interfaces: {GigabitEthernet2/1*
{recovery {5 25} {90 180} {600 100}} }     <----- The bracketed configurations from all WANMON
statements will be included in this line
Aug 8 21:24:14.821: %HA_EM-6-LOG: tm_wanmon.tcl: last_polled_run: 1249760980     <----- Event*
```

```
identifier for exec_count 5
Aug 8 21:24:14.821: %HA_EM-6-LOG: tm_wanmon.tcl: current_polled_run: 1249766654     <-----*
Event identifier for exec_count 6 which is the current process
Aug 8 21:24:14.823: %HA_EM-6-LOG: tm_wanmon.tcl: drift: 5074*
Aug 8 21:24:14.823: %HA_EM-6-LOG: tm_wanmon.tcl: abs_drift: 5074*
Aug 8 21:24:14.823: %HA_EM-6-LOG: tm_wanmon.tcl: drift_thres: 3600*
Aug 8 21:24:14.825: %HA_EM-4-LOG: tm_wanmon.tcl: Clock drift (5074 s), threshold exceeded,*
reseting context
Aug 8 21:24:14.827: %HA_EM-6-LOG: tm_wanmon.tcl: Querying all interface name and states     <--*
--- This output signifies that WANMON is now polling layer 1 and 2 status for all enabled
interfaces
Aug 8 21:24:14.833: %HA_EM-6-LOG: tm_wanmon.tcl: Command: show interface GigabitEthernet2/1 |*
include line protocol     <----- Command send to IOS
Aug 8 21:24:15.047: %HA_EM-6-LOG: tm_wanmon.tcl: Output: GigabitEthernet2/1 is up, line*
protocol is up     <----- Input received from IOS
Aug 8 21:24:15.047: %HA_EM-6-LOG: tm_wanmon.tcl: CGR1000#*
Aug 8 21:24:15.057: %HA_EM-6-LOG: tm_wanmon.tcl: GigabitEthernet2/1 admin_state:1,*
line_state:1     <----- WANMON assigns a binary (0 or 1) value to admin_state and line_state tcl
variables
Aug 8 21:24:15.061: %HA_EM-6-LOG: tm_wanmon.tcl: Processing all interface config*
Aug 8 21:24:15.087: %HA_EM-6-LOG: tm_wanmon.tcl: Querying all IP addresses*
Aug 8 21:24:15.099: %HA_EM-6-LOG: tm_wanmon.tcl: Querying all IPSLA     <----- As IP SlA is not*
attached to this configuration, there is no IP SLA output
Aug 8 21:24:15.105: %HA_EM-6-LOG: tm_wanmon.tcl: Querying all traffic stats*
Aug 8 21:24:15.109: %HA_EM-6-LOG: tm_wanmon.tcl: Processing all interface statistics*
:Aug 8 21:24:15.113: %HA_EM-6-LOG: tm_wanmon.tcl: Interface GigabitEthernet2/1*
Aug 8 21:24:15.119: %HA_EM-6-LOG: tm_wanmon.tcl: GigabitEthernet2/1 lineproto=1 => 1     <-----*
"The script's logic is that if line_state is 1 or greater then consider the status as "up
Aug 8 21:24:15.131: %HA_EM-1-LOG: tm_wanmon.tcl: Detected GigabitEthernet2/1 link is up     <--*
--- Up status confirmed
Aug 8 21:24:15.137: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,admin_state) = 1*
= (Aug 8 21:24:15.137: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,causes*
Aug 8 21:24:15.137: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,checkip) = 0*
Aug 8 21:24:15.139: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,checklist) =*
lineproto
Aug 8 21:24:15.139: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,exists) = 1*
Aug 8 21:24:15.139: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,fullname) =*
GigabitEthernet2/1     <----- This section is where the recovery level timers are shown as well
.as whether or not the interface state is a 0 or 1
Aug 8 21:24:15.139: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,lineproto,state)*
= 1
Aug 8 21:24:15.139: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,recovery,0,max)*
= 25
Aug 8 21:24:15.141: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/1,recovery,0,thresh) = 5
Aug 8 21:24:15.141: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,recovery,1,max)*
= 180
Aug 8 21:24:15.141: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/1,recovery,1,thresh) = 90
Aug 8 21:24:15.143: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,recovery,2,max)*
= 100
Aug 8 21:24:15.143: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/1,recovery,2,thresh) = 600
Aug 8 21:24:15.145: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,state) = 1   *
<----- Once the script has completed, the 'state' value is set to 1 and no recovery actions are
taken
Aug 8 21:24:15.145: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(current_polled_run) = 1249766654   *
<----- This is the event identifier which correlates with exec_count 6
Aug 8 21:24:15.145: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(if_list) = GigabitEthernet2/1*
Aug 8 21:24:15.147: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_saved_ctx(GigabitEthernet2/1,link_events) = {1 {} 1249766655 {Sat Aug 08 21:24:15 UTC
2009}}     <----- exec_count 6 is finished and its historical polling data has been saved
Aug 8 21:24:15.147: %HA_EM-6-LOG: tm_wanmon.tcl: Finished     <----- Regular polling has*
finished
```

```
Aug 8 21:24:15.159: %HA_EM-6-LOG: tm_wanmon.tcl: context_save succeeded     <----- The process*
                                                 has been statefully ended without error
```

مشكلة شائعة أثناء تشغيل البرنامج النصي

خطأ في بناء الجملة

قد تنشأ حالة تعتقد فيها أن WANMON يعمل عندما يتم تكوينه، ولكن في الواقع ليس له أي تأثير. على سبيل المثال، في بيئة لم يعمل فيها البرنامج النصي ل WANmon بشكل صحيح، تم ملاحظة مخرج تصحيح الأخطاء هذا:

```
Aug 8 21:44:14.487: %HA_EM-6-LOG: tm_wanmon.tcl: Command: enable*
:Aug 8 21:44:14.601: %HA_EM-6-LOG: tm_wanmon.tcl: Output*
Aug 8 21:44:14.601: %HA_EM-6-LOG: tm_wanmon.tcl: CGR1000#*
Aug 8 21:44:14.617: %HA_EM-6-LOG: tm_wanmon.tcl: context_retrieve succeeded*
Aug 8 21:44:14.617: %HA_EM-6-LOG: tm_wanmon.tcl: last_run: 1249767255*
Aug 8 21:44:14.617: %HA_EM-6-LOG: tm_wanmon.tcl: exec_count: 1*
Aug 8 21:44:14.619: %HA_EM-6-LOG: tm_wanmon.tcl: GigabitEthernet2/1,link_events: {1 {}*
{{1249766655 {Sat Aug 08 21:24:15 UTC 2009
Aug 8 21:44:14.619: %HA_EM-6-LOG: tm_wanmon.tcl: last_polled_run: 1249767255*
Aug 8 21:44:14.627: %HA_EM-6-LOG: tm_wanmon.tcl: Starting: Received event polltimer*
{} :Aug 8 21:44:14.627: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_name Value*
Aug 8 21:44:14.629: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_pub_sec Value: 1249767854*
Aug 8 21:44:14.629: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_type Value: 21*
Aug 8 21:44:14.629: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_remain Value: 600.000*
Aug 8 21:44:14.629: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_severity Value: severity-normal*
Aug 8 21:44:14.629: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_id Value: 3*
Aug 8 21:44:14.631: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_type_string Value: timer watchdog*
Aug 8 21:44:14.631: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_time_sec Value: 3458756654*
Aug 8 21:44:14.631: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_type Value: watchdog*
Aug 8 21:44:14.633: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_remain_msec Value: 0*
Aug 8 21:44:14.633: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_remain_sec Value: 600*
Aug 8 21:44:14.633: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_time_msec Value: 45*
Aug 8 21:44:14.635: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_pub_time Value: 1249767854.045*
Aug 8 21:44:14.635: %HA_EM-6-LOG: tm_wanmon.tcl: Name: job_id Value: 8*
Aug 8 21:44:14.635: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_pub_msec Value: 45*
Aug 8 21:44:14.635: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_time Value: 3458756654.045*
Aug 8 21:44:14.635: %HA_EM-6-LOG: tm_wanmon.tcl: Exec count: 2*
Aug 8 21:44:14.639: %HA_EM-6-LOG: tm_wanmon.tcl: All interface vars wanmon_if_list1*
Aug 8 21:44:14.641: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_interfaces: {gig 2/1 {recovery {5 25}*
{ {{{90 180} {400 100
Aug 8 21:44:14.649: %HA_EM-6-LOG: tm_wanmon.tcl: last_polled_run: 1249767255*
Aug 8 21:44:14.649: %HA_EM-6-LOG: tm_wanmon.tcl: current_polled_run: 1249767854*
Aug 8 21:44:14.651: %HA_EM-6-LOG: tm_wanmon.tcl: drift: -1*
Aug 8 21:44:14.651: %HA_EM-6-LOG: tm_wanmon.tcl: abs_drift: 1*
Aug 8 21:44:14.651: %HA_EM-6-LOG: tm_wanmon.tcl: drift_thres: 3600*
Aug 8 21:44:14.655: %HA_EM-6-LOG: tm_wanmon.tcl: Querying all interface name and states*
Aug 8 21:44:14.659: %HA_EM-6-LOG: tm_wanmon.tcl: Command: show interface gig | include line*
protocol     <----- Here you can see the command which has been sent to IOS which indicates that
         'show interface gig' instead of 'show interface gig1/2' has been sent
:Aug 8 21:44:14.889: %HA_EM-6-LOG: tm_wanmon.tcl: Output*
Aug 8 21:44:14.889: %HA_EM-6-LOG: tm_wanmon.tcl: show interface gig | include line protocol*
^ :Aug 8 21:44:14.889: %HA_EM-6-LOG: tm_wanmon.tcl*
.Aug 8 21:44:14.889: %HA_EM-6-LOG: tm_wanmon.tcl: % Invalid input detected at '^' marker*
:Aug 8 21:44:14.889: %HA_EM-6-LOG: tm_wanmon.tcl*
Aug 8 21:44:14.889: %HA_EM-6-LOG: tm_wanmon.tcl: CGR1000#*
Aug 8 21:44:14.895: %HA_EM-1-LOG: tm_wanmon.tcl: Configuration error: Interface gig does not*
      exist. Please check configuration.     <----- If there is a syntax error in the WANMON
   statements, you will see "Configuration error" in the output following the raw session output
Aug 8 21:44:14.899: %HA_EM-6-LOG: tm_wanmon.tcl: Processing all interface config*
Aug 8 21:44:14.915: %HA_EM-6-LOG: tm_wanmon.tcl: context_save succeeded*
```

طبقا لهذا الإخراج، لا يتم تشغيل البرنامج النصي بنجاح نظرا لوجود مسافة بين اسم الواجهة وأرقامها. الأمر الذي تسبب

في هذا الخطأ هو:

```
{ {{event manager environment wanmon_if_list1 {gig 2/1 {recovery {5 25} {90 180} {400 100
```

يجب أن يكون التكوين الصحيح:

```
{ {{event manager environment wanmon_if_list1 {gig2/1 {recovery {5 25} {90 180} {400 100
```

إتفاقية مستوى خدمة بروتوكول الإنترنت (IPSLA) للأسفل

إذا استطلع WANMON الواجهات المتعددة مع IPSLA.

عمليات IPSLA، قد يكون من الصعب تحديد مكان الفشل على الفور. يوضح إخراج تصحيح الأخطاء هذا ما يحدث عندما يتم مراقبة واجهتين بواسطة WANmon مع IP SLA على كل واجهة:

```
Aug 9 19:55:36.433: %HA_EM-6-LOG: tm_wanmon.tcl: Command: enable*
:Aug 9 19:55:36.547: %HA_EM-6-LOG: tm_wanmon.tcl: Output*
Aug 9 19:55:36.547: %HA_EM-6-LOG: tm_wanmon.tcl: CGR1000#*
Aug 9 19:55:36.565: %HA_EM-6-LOG: tm_wanmon.tcl: context_retrieve succeeded*
Aug 9 19:55:36.565: %HA_EM-6-LOG: tm_wanmon.tcl: last_run: 1249847563*
Aug 9 19:55:36.565: %HA_EM-6-LOG: tm_wanmon.tcl: exec_count: 5*
Aug 9 19:55:36.565: %HA_EM-6-LOG: tm_wanmon.tcl: GigabitEthernet2/1,link_events: {0 ipsla*
{{1249847563 {Sun Aug 09 19:52:43 UTC 2009}} {1 {} 1249847478 {Sun Aug 09 19:51:18 UTC 2009
Aug 9 19:55:36.567: %HA_EM-6-LOG: tm_wanmon.tcl: GigabitEthernet2/2,link_events: {0 lineproto*
{{1249847478 {Sun Aug 09 19:51:18 UTC 2009
Aug 9 19:55:36.567: %HA_EM-6-LOG: tm_wanmon.tcl: last_polled_run: 1249847136*
Aug 9 19:55:36.575: %HA_EM-6-LOG: tm_wanmon.tcl: Starting: Received event polltimer*
{} :Aug 9 19:55:36.577: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_name Value*
Aug 9 19:55:36.577: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_pub_sec Value: 1249847735*
Aug 9 19:55:36.577: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_type Value: 21*
Aug 9 19:55:36.577: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_remain Value: 600.000*
Aug 9 19:55:36.579: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_severity Value: severity-normal*
Aug 9 19:55:36.579: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_id Value: 1*
Aug 9 19:55:36.579: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_type_string Value: timer watchdog*
Aug 9 19:55:36.581: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_time_sec Value: 3458836535*
Aug 9 19:55:36.581: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_type Value: watchdog*
Aug 9 19:55:36.581: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_remain_msec Value: 0*
Aug 9 19:55:36.583: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_remain_sec Value: 600*
Aug 9 19:55:36.583: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_time_msec Value: 908*
Aug 9 19:55:36.583: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_pub_time Value: 1249847735.907*
Aug 9 19:55:36.585: %HA_EM-6-LOG: tm_wanmon.tcl: Name: job_id Value: 6*
Aug 9 19:55:36.585: %HA_EM-6-LOG: tm_wanmon.tcl: Name: event_pub_msec Value: 907*
Aug 9 19:55:36.585: %HA_EM-6-LOG: tm_wanmon.tcl: Name: timer_time Value: 3458836535.908*
Aug 9 19:55:36.585: %HA_EM-6-LOG: tm_wanmon.tcl: Exec count: 6     <----- The new polling*
interval has begun
Aug 9 19:55:36.591: %HA_EM-6-LOG: tm_wanmon.tcl: All interface vars wanmon_if_list1*
wanmon_if_list3     <----- Both lists are included in this output
Aug 9 19:55:36.593: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_interfaces: {GigabitEthernet2/1 {ipsla*
1} } {GigabitEthernet2/2 {ipsla 2} }     <----- Both bracketed configurations are included in
this output
Aug 9 19:55:36.601: %HA_EM-6-LOG: tm_wanmon.tcl: last_polled_run: 1249847136*
Aug 9 19:55:36.603: %HA_EM-6-LOG: tm_wanmon.tcl: current_polled_run: 1249847736*
Aug 9 19:55:36.603: %HA_EM-6-LOG: tm_wanmon.tcl: drift: 0*
Aug 9 19:55:36.603: %HA_EM-6-LOG: tm_wanmon.tcl: abs_drift: 0*
Aug 9 19:55:36.603: %HA_EM-6-LOG: tm_wanmon.tcl: drift_thres: 3600*
Aug 9 19:55:36.605: %HA_EM-6-LOG: tm_wanmon.tcl: Querying all interface name and states     <--*
--- WANMON sends show commands to IOS for both enabled interfaces
Aug 9 19:55:36.613: %HA_EM-6-LOG: tm_wanmon.tcl: Command: show interface GigabitEthernet2/1 |*
include line protocol
```

```
Aug 9 19:55:36.825: %HA_EM-6-LOG: tm_wanmon.tcl: Output: GigabitEthernet2/1 is up, line*
protocol is up
Aug 9 19:55:36.825: %HA_EM-6-LOG: tm_wanmon.tcl: CGR1000#*
Aug 9 19:55:36.835: %HA_EM-6-LOG: tm_wanmon.tcl: GigabitEthernet2/1 admin_state:1,*
line_state:1     <----- GigabitEthernet2/1 is up/up so it sets admin_state and line_state to 1
Aug 9 19:55:36.835: %HA_EM-6-LOG: tm_wanmon.tcl: Command: show interface GigabitEthernet2/2 |*
include line protocol
Aug 9 19:55:37.047: %HA_EM-6-LOG: tm_wanmon.tcl: Output: GigabitEthernet2/2 is down, line*
protocol is down
Aug 9 19:55:37.047: %HA_EM-6-LOG: tm_wanmon.tcl: CGR1000#*
Aug 9 19:55:37.055: %HA_EM-6-LOG: tm_wanmon.tcl: GigabitEthernet2/2 admin_state:1,*
line_state:0     <----- GigabitEthernet2/2 is down/down but admin_state is still set to 1
because it is administratively enabled. Since layer 2 is down, line_state is set to 0
Aug 9 19:55:37.059: %HA_EM-6-LOG: tm_wanmon.tcl: Processing all interface config*
Aug 9 19:55:37.083: %HA_EM-6-LOG: tm_wanmon.tcl: Querying all IP addresses*
Aug 9 19:55:37.095: %HA_EM-6-LOG: tm_wanmon.tcl: Querying all IPSLA     <----- Due to the fact*
that IP SLA operations are configured for both interfaces, WANMON now sends show commands to
check IP SLA status
Aug 9 19:55:37.097: %HA_EM-6-LOG: tm_wanmon.tcl: Querying IP SLA for GigabitEthernet2/1*
Aug 9 19:55:37.099: %HA_EM-6-LOG: tm_wanmon.tcl: Command: show ip sla statistics 1 | include*
Latest operation return code:     <----- IP SLA 1 is queried
Aug 9 19:55:37.309: %HA_EM-6-LOG: tm_wanmon.tcl: Output: Latest operation return code: Timeout *
<----- The state is down because the last return code for the icmp-echo IP SLA process is
Timeout
Aug 9 19:55:37.309: %HA_EM-6-LOG: tm_wanmon.tcl: CGR1000#*
Aug 9 19:55:37.311: %HA_EM-6-LOG: tm_wanmon.tcl: GigabitEthernet2/1 ipsla: 0     <----- The*
ipsla value for Gig2/1 is now set to 0 because the state is down
Aug 9 19:55:37.313: %HA_EM-6-LOG: tm_wanmon.tcl: Querying IP SLA for GigabitEthernet2/2*
Aug 9 19:55:37.313: %HA_EM-6-LOG: tm_wanmon.tcl: Command: show ip sla statistics 2 | include*
:Latest operation return code
Aug 9 19:55:37.527: %HA_EM-6-LOG: tm_wanmon.tcl: Output: Latest operation return code: OK   *
<----- IP SLA operation 2 is up, the remote address is reachable
Aug 9 19:55:37.527: %HA_EM-6-LOG: tm_wanmon.tcl: CGR1000#*
Aug 9 19:55:37.529: %HA_EM-6-LOG: tm_wanmon.tcl: GigabitEthernet2/2 ipsla: 1     <----- The*
ipsla value for Gig2/2 is now set to 1 because the state is up
Aug 9 19:55:37.535: %HA_EM-6-LOG: tm_wanmon.tcl: Querying all traffic stats*
Aug 9 19:55:37.539: %HA_EM-6-LOG: tm_wanmon.tcl: Processing all interface statistics*
:Aug 9 19:55:37.543: %HA_EM-6-LOG: tm_wanmon.tcl: Interface GigabitEthernet2/1*
:Aug 9 19:55:37.543: %HA_EM-6-LOG: tm_wanmon.tcl: Interface GigabitEthernet2/2*
Aug 9 19:55:37.551: %HA_EM-6-LOG: tm_wanmon.tcl: GigabitEthernet2/1 lineproto=1 ipsla=0 => 0   *
<----- WANMON now checks for 2 dependencies in order to determine state. For interface Gig2/1
.the IP SLA state is down even though the interface is up/up
Aug 9 19:55:37.555: %HA_EM-6-LOG: tm_wanmon.tcl: GigabitEthernet2/2 lineproto=0 ipsla=1 => 0   *
<----- For Gig2/2, the IP SLA state is up but the interface is down
Aug 9 19:55:37.589: %HA_EM-6-LOG: tm_wanmon.tcl: Interface GigabitEthernet2/1, Level 0 recovery*
within 10 min
Aug 9 19:55:37.593: %HA_EM-6-LOG: tm_wanmon.tcl: Interface GigabitEthernet2/1, Level 1 recovery*
within 60 min
Aug 9 19:55:37.595: %HA_EM-6-LOG: tm_wanmon.tcl: Interface GigabitEthernet2/1, Level 2 recovery*
within 480 min
Aug 9 19:55:37.597: %HA_EM-2-LOG: tm_wanmon.tcl: Interface GigabitEthernet2/1: Level 0 recovery*
in 10 min
Aug 9 19:55:37.603: %HA_EM-6-LOG: tm_wanmon.tcl: Interface GigabitEthernet2/2, Level 0 recovery*
within 10 min
Aug 9 19:55:37.605: %HA_EM-6-LOG: tm_wanmon.tcl: Interface GigabitEthernet2/2, Level 1 recovery*
within 60 min
Aug 9 19:55:37.609: %HA_EM-6-LOG: tm_wanmon.tcl: Interface GigabitEthernet2/2, Level 2 recovery*
within 480 min
Aug 9 19:55:37.609: %HA_EM-2-LOG: tm_wanmon.tcl: Interface GigabitEthernet2/2: Level 0 recovery*
in 10 min
Aug 9 19:55:37.613: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,admin_state) = 1*
Aug 9 19:55:37.615: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,causes) = ipsla*
Aug 9 19:55:37.615: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,check_ipsla) = 1*
Aug 9 19:55:37.615: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,checkip) = 0*
Aug 9 19:55:37.615: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,checklist) =*
lineproto ipsla
Aug 9 19:55:37.617: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,exists) = 1*
Aug 9 19:55:37.617: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,fullname) =*
GigabitEthernet2/1
Aug 9 19:55:37.617: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,ipsla,state) = 0*
Aug 9 19:55:37.619: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,lineproto,state)*
= 1
Aug 9 19:55:37.619: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,recovery,0,max)*
= 10
Aug 9 19:55:37.619: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/1,recovery,0,thresh) = 10
Aug 9 19:55:37.621: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,recovery,1,max)*
= 60
Aug 9 19:55:37.621: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/1,recovery,1,thresh) = 60
Aug 9 19:55:37.621: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,recovery,2,max)*
= 60
Aug 9 19:55:37.621: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/1,recovery,2,thresh) = 480
Aug 9 19:55:37.623: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/1,recovery,next,level) = 0
Aug 9 19:55:37.623: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/1,recovery,next,repeat) = 0
Aug 9 19:55:37.623: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/1,recovery,next,time_left) = 600
Aug 9 19:55:37.625: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,slaid) = 1     *
<----- This output gives you the SLA ID (1) of the process bound to the Gig2/1 interface
Aug 9 19:55:37.625: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/1,state) = 0   *
<----- The state has been marked 0 or down due to the fact that check_ipsla value is 0 even
though the interface is up and line_state is 1
Aug 9 19:55:37.627: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/2,admin_state) = 1*
Aug 9 19:55:37.627: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/2,causes) =*
lineproto
Aug 9 19:55:37.627: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/2,check_ipsla) = 1*
Aug 9 19:55:37.629: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/2,checkip) = 0*
Aug 9 19:55:37.629: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/2,checklist) =*
lineproto ipsla
Aug 9 19:55:37.629: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/2,exists) = 1*
Aug 9 19:55:37.631: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/2,fullname) =*
GigabitEthernet2/2
Aug 9 19:55:37.631: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/2,ipsla,state) = 1*
Aug 9 19:55:37.631: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/2,lineproto,state)*
= 0
Aug 9 19:55:37.631: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/2,recovery,0,max)*
= 10
Aug 9 19:55:37.633: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/2,recovery,0,thresh) = 10
Aug 9 19:55:37.633: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/2,recovery,1,max)*
= 60
Aug 9 19:55:37.635: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/2,recovery,1,thresh) = 60
Aug 9 19:55:37.635: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/2,recovery,2,max)*
= 60
Aug 9 19:55:37.635: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/2,recovery,2,thresh) = 480
Aug 9 19:55:37.637: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/2,recovery,next,level) = 0
Aug 9 19:55:37.637: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/2,recovery,next,repeat) = 0
Aug 9 19:55:37.637: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_ctx(GigabitEthernet2/2,recovery,next,time_left) = 600
Aug 9 19:55:37.639: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/2,slaid) = 2*
Aug 9 19:55:37.639: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(GigabitEthernet2/2,state) = 0   *
```

```
<----- Gig2/2 has been marked 0 or down due to the fact that the line_state is 0, despite the
fact that the IP SLA operation is still UP and the check_ipsla value is 1
Aug 9 19:55:37.639: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(current_polled_run) = 1249847736*
Aug 9 19:55:37.639: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_ctx(if_list) = GigabitEthernet2/1*
GigabitEthernet2/2
Aug 9 19:55:37.641: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_saved_ctx(GigabitEthernet2/1,link_events) = {0 ipsla 1249847563 {Sun Aug 09 19:52:43 UTC
{{2009}} {1 {} 1249847478 {Sun Aug 09 19:51:18 UTC 2009
Aug 9 19:55:37.641: %HA_EM-6-LOG: tm_wanmon.tcl:*
wanmon_saved_ctx(GigabitEthernet2/2,link_events) = {0 lineproto 1249847478 {Sun Aug 09 19:51:18
{{UTC 2009
Aug 9 19:55:37.641: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_saved_ctx(exec_count) = 6*
Aug 9 19:55:37.641: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_saved_ctx(last_polled_run) = 1249847136*
Aug 9 19:55:37.643: %HA_EM-6-LOG: tm_wanmon.tcl: wanmon_saved_ctx(last_run) = 1249847563*
Aug 9 19:55:37.643: %HA_EM-6-LOG: tm_wanmon.tcl: Finished*
Aug 9 19:55:37.661: %HA_EM-6-LOG: tm_wanmon.tcl: context_save succeeded*
```

معلومات ذات صلة

- http://www.cisco.com/c/en/us/td/docs/routers/connectedgrid/cgr1000/ios/software/15_4_1_cg/WANMon.html
- http://www.cisco.com/c/en/us/td/docs/routers/connectedgrid/cgr1000/1_0/software/configuration/guide/unicast/Unicast_Book/l3_backhaul_cgr1000.html#wp998807
- الدعم التقني والمستندات - Cisco Systems

حول هذه الترجمة

ترجمت Cisco هذا المستند باستخدام مجموعة من التقنيات الآلية والبشرية لتقديم محتوى دعم للمستخدمين في جميع أنحاء العالم بلغتهم الخاصة. يُرجى ملاحظة أن أفضل ترجمة آلية لن تكون دقيقة كما هو الحال مع الترجمة الاحترافية التي يقدمها مترجم محترف. تخلي Cisco Systems مسئوليتها عن دقة هذه الترجمات وتُوصي بالرجوع دائمًا إلى المستند الإنجليزي الأصلي (الرابط متوفر).