基於自動化維運之網路管理與監控系統

林孟璋
國家實驗研究院 國家高速網路與計算中心
0303813@narlabs.org.tw

摘要

骨幹網路與資料中心的成立，倍增的網路頻寬與資訊量暴增的環境下，網路管理自動化成為維運的新趨勢，透過模組 Netmiko 開發以網路維運的自動化程式，藉以建立第二層虛擬私有網路(Layer-2 VPN)，減少工程師為每一台路由器鍵入指令的繁瑣工作，有效降低出錯機率。除此之外，網管介面異常偵測連結自動化系統，當介面發生 Flapping 時，自動登入設備，檢查是否因介面異常造成路由的頻繁異動，進而評估其影響範圍與等級，此外，更擴展自動化程式的呼叫範圍與方式，讓不同的網路監控程式可跨系統、跨網段使用此系統。

關鍵詞：自動化、Netmiko、Flapping、 Layer2 VPN。

1. 前言

TWAREN 是高品質學術研究網路的簡稱，自建置100G 高頻寬骨幹網路以來，維運團隊除負責國內研究網路的維運與規劃之外，也陸續更換國際連網設備與成立福爾摩沙開放交換中心(FOX)[1]，為了維運整個骨幹網路與交換中心，設備的多樣性且複雜連結的情況下，工程師不僅維運不同品牌的設備，也需要熟悉不同的操作業系統指令，因此以 SSH(Secure Shell Protocol)登入設備執行指令成為最自動化最主要方式之一，透過 SSH 登入設備執行自動化腳本，並將執行腳本後的結果回傳給管理端，藉以檢查設備、確認網路狀態與確認除錯方向。除此之外，自動化軟體也可以針對設備下達擷取資料的指令，比起大部分網管系統所使用 SNMP(Simple Network Management Protocol)，能獲得更多樣性的資料，對於網管資料的取得更多樣性，這樣監控標的可以更多更豐富，例如取得 BGP 路由前綴(Prefixes)筆數，這想監控標的很難以 SNMP 方式取得，但對於跟 TWAREN 對連的研究網路，路由筆數的異動佔了很重要的角色，也因此自動化介入成了必要的手段之一。

VPN(虛擬私有網路)是 TWAREN 重要的服務項目之一，二線工程師接到 Layer-2 VPN 的服務需求時，針對有著相同設定指令且每台路由器都要設定的特色，使用自動化可以減少工程師在設定時鍵錯指令機率，對於輔助工程師工作上佔有一席重要地位。

但不僅是工程師對骨幹設備上的設定，如外部的網管監控程式需要呼叫，自動化程式也可以幫忙除錯，一般來說取得帳號藉以登入網路設備，如帳號控管不易，有可能造成資訊安全的疑慮，因此可以透過呼叫外部指令方式做認證，被呼叫程式再以 ACL(Access Control List)做呼叫端管控，達成自動化程式檢查設備的目的，一來達到減少誤報警的發生，二來透過程式再次確認，達成減少人為登入執行指令並檢視結果的作為。

2. 研究動機

網路管理系統在網路維運上佔有舉足輕重的角色，TWAREN 骨幹網路服務中，如果少了網管監控，發生網路斷線，事後才知道障礙發生的窘境，且網路障礙事件發生後，處理流程與工作日誌無法記錄，網路工程師總是疲於奔命於解決相同的障礙，缺乏事件管理與標準作業流程的情況下，徒增維運人員的困擾，因此我們以自動化的方式增進網路管理上的效率，網路維運中心人員不必再行登入設備檢查網路狀態，事件直接觸發自動化程式，檢查設備狀態資訊，即時資訊回報給二線工程師，作為判斷與處置的依據，以達到有效管理網路的優點。

Layer-2 VPN(虛擬私有網路)是 TWAREN 主要的服務之一，L2 VPN 提供不同地理位置之間透通的網路，猶如讓在不同地方的使用者如同連到同一台的交換器(Switch)上，從成本效益與安全性而言，L2-VPN 成為比專線更好的方案之一，然而建立 L2-VPN 的透通連接，以骨幹網路的架構而言，需要在各個區網中心(GigaPops)的路由器上鍵入指令，完成建立虛擬網路。因此我們透過自動化軟體，將預先設定好選項提供給工程師選擇，只要填入要在哪一台設備並鍵入關鍵指令，就可以直接部屬到想要的哪台設備上，達到快速且正確建立 VPN 工作，有助於減少工程師登入設備鍵錯指令的窘境。

網路異常偵測是網管系統的重要一環，透過獲取路由器的系統日誌(System logs)，將其匯入大數據資料庫進行分析，如果介面頻繁的啟動(Up)或下線(Down)，通常我們稱之為設備介面震盪(Flapping)的狀況發生，有可能影響網路對外連線，日後並造成介面永久損壞，在這個情況，系統應該即時的發出告警信件，提醒工程師，設備即將損壞且需進行硬體更換，這時我們應該謹慎些，登入設備透過自動化程式，檢查這個介面對於連線的影響，通常影響到的是 IGP(Inter Gateway Protocol)，這是內部路由訊息的一種協議，透過位於同一自治編號(Autonomous Number)交換路由訊息的一個協議，讓封包數據可以順利找到最佳路徑，一般來說是跑 OSPF，然而震盪是否會影響到 IGP 這就需要工程師登入路由器下達指令確認。透過自動化程式登入發生震盪的設備，下達檢查指令，予以佐證，達成查修。

自動化系統開發更重要的議題有關於，該如何傳遞想要的指令給自動化系統，讓自動化系統去執行想要的指令，因此我們想到 RESTful API (Representational State Transfer API)的一個呼叫環境，RESTfull 並非一種架構或是協議，而是設計 Web API 的方式，利用 HTTP 協議的特性來實踐數據交換。這樣使用自動化程式就可以透過 HTTP 方式呼叫，透過輸入想要執行的指令與哪台設備就可以回傳結果，易於各種不同的資訊系統作為資料交換與傳遞。

3. 系統架構

在規劃網路自動化架構時，我們希望自動化系統都可以給予網管系統呼叫使用，朝 HTTP 通訊協定當作資料交換的管道，不管是透過程式呼叫方式亦或者瀏覽器操作，都可藉由認證方式橋接自動化系統。架構由成如圖1。

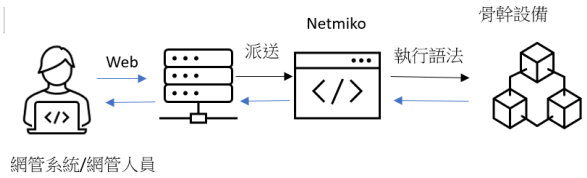

圖 1 架構圖

3.1 Netmiko

Netmiko[2]是一套以 Python 為基底的自動化開發模組，用來執行登入設備與執行指令之用，其前身為 Paramiko，在開發者陸續開放源碼的趨勢下，Netmiko 模組化軟體孕育而生，Netmiko 更專注於不同網路設備的指令互動模式，此為選擇以此作為自動化模組最主要依據，以此因應 TWAREN 骨幹網路不同廠商的設備加入，可以是自動化的好工具之一。開發端的 Python 版本升級到3.6版本以上即可使用此模組，其安裝程式如下：

```
python3.8 -m pip install --upgrade pip
pip3 install netmiko
```

我們以登入設備並下顯示介面 gi0/0/0/0的範例來說明，使用此函示庫的步驟依序為登入設備、執行指令、切斷連線步驟。使用 Netmiko API 的程式片段如下「第(1)式~(5)式」。

```
from netmiko import ConnectHandler                    (1)
 device=
ConnectHandler(device_type='cisco_ios',ip='192.168
.248.249',user='user',password='test')                (2)
output= device.send_command("show interface
gi0/0/0/0 ")                                          (3)
print(output)                                         (4)
device.disconnect()                                   (5)
```

第一行透過 import 這個關鍵字載入模組。第二行為連接設備函數 ConnectHandler，參數依序為設備類型 cisco_ios、設備 IP、帳號、密碼，並將回傳值傳給變數 device 這個操作者(Handler)，第一個參數 cisco_ios 為 IOS 作業系統，在商業運轉環境下可透過 SSH 登入作業，依不同廠商的設備在這個關鍵參數做修改。第三行為透過操作者做指令的下達，在此下達列出有關介面 gi0/0/0/0的描述。第四行將結果予以印出。第五行結束設備連接。即可在程式輸出看到如圖2結果。

```
Tue Aug 16 05:57:10.318 UTC
GigabitEthernet0/0/0/0 is up, line protocol is up
  Interface state transitions: 1
  Hardware is GigabitEthernet, address is 0c43.c3c0.0001 (bia 0c43.c3c0.0001)
  Internet address is 10.1.1.2/24
  MTU 1514 bytes, BW 1000000 Kbit (Max: 1000000 Kbit)
     reliability 255/255, txload 0/255, rxload 0/255
  Encapsulation ARPA,
  Full-duplex, 1000Mb/s, unknown, link type is force-up
  output flow control is off, input flow control is off
  Carrier delay (up) is 10 msec
  loopback not set,
  Last link flapped 00:40:53
  ARP type ARPA, ARP timeout 04:00:00
  Last input never, output 00:40:50
  Last clearing of "show interface" counters never
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     0 packets input, 0 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 0 multicast packets
              0 runts, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     1 packets output, 42 bytes, 0 total output drops
     Output 1 broadcast packets, 0 multicast packets
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
     1 carrier transitions
```

圖 2 輸出結果

除了以 Python 可以利用處理字串函數找出想要的資訊，還透過名為 TextFSM[3]的模組將顯示內容有序的以 JSON 格式化輸出，方便我們過濾與取出想要的資料，JSON 為大家熟悉的資料交換格式，對於日後這些設備執行後，將回傳的資料後續寫入大資料庫進行分析，可以是說非常便利的模組。

3.2 外部呼叫

Python 本身不是用來直接執行網頁上的程式語言，因為它偏向於後端執行腳本的角色，當自動化程式登入路由器等重要設備時，登入帳號資訊將植入程式中，顧及資訊安全的考量，不允許直接被存取，而是給另外一支程式呼叫使用。我們以另外一支程式做呼叫當輔助，這程式開發可以透過個人熟悉的 Web 程式語言來進行，例如 PHP 這類的網頁應用程式，我們以它撰寫對外介面的功能。以下程式碼就是網頁應用程式 PHP 執行 Netmiko 其中指令，如下「第(6)式」、「(7)式」。

```
$execution_cmd='/usr/bin/python3.8
/autom/netmiko/netmiko_non_textfsm.py    '.$host.'
'."".$rt_cmd."";                                      (6)
exec ($execution_cmd,$out,$status);                   (7)
```

執行後的結果儲存於$out 變數，除了提供後續使用之外，PHP 有許多正規表示式(Regular Expression)的功能，讓網頁僅呈現符合字串的內容。

通常網管程式中呼叫自動化軟體時，是以 Web

請求當作最主要溝通管道，立即想到 CURL [4]這套強大的命令列工具來存取網頁，CURL 軟體提供針對請求的多樣性，包含 GET、POST、PUT 的請求方法，以及使用表頭(Header)帶入 JSON 格式的方式針對伺服器發出請求。呼叫自動化程式時只要填入必須的 URL，例如以下的執行程式碼:

curl "http://192.168.3.x/netmiko/get_ospf7539.php?host=TWAREN-TN-ASR9006-01&rt_cmd=show%20ospf%207539%20neighbor%20TenGigE0/1/1/6" (8)

「第(8)式」是針對 Netmiko 開發的取得 IGP 狀態的外部呼叫指令。透過傳遞參數 host、rt_cmd，這兩個參數分別代表登入哪台設備、執行什麼樣的指令，我們分別給予 TWAREN-TN-ASR9006-01 這台路由器名稱，執行「show ospf 7539 neighbor TenGigE0/1/1/6」的指令，因為指令之間有空白的關係，需要以"%20"代替空白，或以 URL 編碼(URL Encoding)的方式將 URL 傳給 Curl 執行。執行結果如圖3，說明這個鄰居(Neighbor)已經持續連線3天。

```
Fri Jul 26 16:48:15.853 cst
Neighbors for OSPF 7539

Neighbor ID     Pri   State           Dead Time   Address         Interface
211.79.48.134   128   FULL/  -        00:00:38    211.79.48.114   TenGigE0/1/1/6
    Neighbor is up for 3d01h
```

圖3 呼叫自動化之結果

4. 案例研究

自動化程式在網路設備上的管理非常多元性，本系統藉由外部呼叫方式，執行自動化的腳本，驗證網管異常偵測系統的正確性，減少工程師登入設備的執行動作。本篇文章舉兩個案例分別為，建立 Layer-2 VPN 的自動化與介面異常偵測的自動化驗證，此兩個案例是以自動化程式來完成，但觸發點卻有所不同，Layer-2 VPN 是以工程師觀點，接到客戶需求，以設定 VPN 為出發點，填入必要欄位，完成自動化設定所需工作;介面異常偵測是以事件觸發為觀點，當網管事件發生時，直接觸發自動化程式，檢查設備狀態，藉此驗證其影響程度與範圍，介面異常往往影響網路品質，與日後更換介面的依據，從介面震盪與失效皆可透過自動化即時檢查並驗證。兩案例在執行 TWAREN 骨幹維運與管理上，皆有所幫助。

4.1 建立 Layer-2 VPN

Layer-2 VPN 以邏輯的觀點來看，在不同地區的客戶連接在一台虛擬的交換器上，透過 Layer-2 VPN 的指令，在每台設備上針對可用的介面(Interface)、設定橋接域(Bridge Domain)、虛擬電路識別符號(Virtual Circuit ID,VC ID)。工程師需要在各台設備上，例行性的使用相同指令，在閒置介面上設定一個共同的橋接域名稱，與一個 VC ID，這樣才可以將同屬於一個橋接域的客戶彼此交換流量，但其中一台設定不正確，網路即不連通。因此針對相同設備多行指令的工作，就交由自動化程式完成。

我們設計了路由器上可用介面的查詢網頁，以登入設備執行自動化程式，讓工程師知道哪台設備那個介面尚可使用，例如，想要知道在這台機器尚可使用介面，或已經知道介面名稱，欲查詢介面使用狀態，亦或者在這台設備上的橋接域有沒有被使用，即可透過如圖4之介面得知。

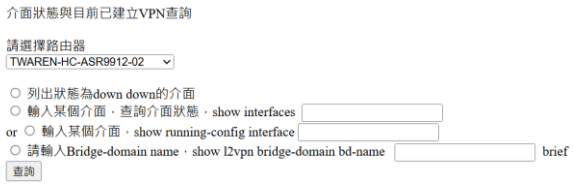

圖4 查詢介面狀態

例如我們選取最上方的路由器名稱 TWAREN-HC-ASR9912-02，填入要查詢的介面名，例如，GigabitEthernet0/9/0/9，即可得到介面使用資訊如下圖5。

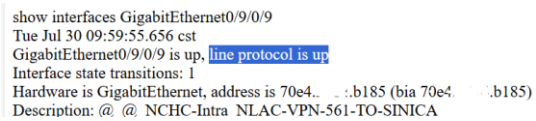

圖5 自動化執行結果

如果具備了可用介面，與制訂好的一組橋接域名稱與 VC ID，就可以將這些資訊填在開發好的網頁資訊欄中，之後進入新增 VPN 的確認步驟，新增時會顯示自動化即將執行的所有指令，給予工程師再次確認，如無誤即派送至設備予以執行。新增與確認執行指令，如圖 6、圖7。

圖 6 新增 VPN 所需必要欄位

自動化在這些路由器上自動執行這些指令，順利生效於各台路由器上，減少工程師鍵入指令的程序。除此之外，由於建立 VPN 指令相較於讀取路由器設定的權限還來的高，除了我們於登入時設計了身份的驗證之外，對於操作的人員動作也予

以記錄成日誌，以利稽核，甚至到自動化程式登入時驗證帳號也予以紀錄，有以此自動化程式運作的帳號時也會發信通知認證管理員，確認是否真有其動作，做為資安的防護屏障。

設定如下

```
conf t
interface GigabitEthernet0/6/0/5
 description @_@_TWAREN-NDHU-ASR9010-01_GigabitEthernet0/6/0/5_OLA-2
 no shutdown
interface GigabitEthernet0/6/0/5.1 l2transport
 description @_@_TWAREN-NDHU-ASR9010-01_GigabitEthernet0/6/0/5_OLA-2
 encapsulation untagged
l2vpn
 bridge group CUS
  bridge-domain OLA-2
   interface GigabitEthernet0/6/0/5.1
   vfi OLA-2
    vpn-id 753
    autodiscovery bgp
     rd 7539:vpn-id 753
     route-target 7539:vpn-id 753
     signaling-protocol ldp
```

```
conf t
interface GigabitEthernet0/9/0/12
 description @_@_TWAREN-HC-ASR9912-02_GigabitEthernet0/9/0/12_OLA-2
 no shutdown
interface GigabitEthernet0/9/0/12.1 l2transport
 description @_@_TWAREN-HC-ASR9912-02_GigabitEthernet0/9/0/12_OLA-2
 encapsulation untagged
l2vpn
 bridge group CUS
  bridge-domain OLA-2
   interface GigabitEthernet0/9/0/12.1
   vfi OLA-2
    vpn-id 753
    autodiscovery bgp
     rd 7539:vpn-id 753
     route-target 7539:vpn-id 753
     signaling-protocol ldp
```

```
conf t
interface GigabitEthernet0/9/0/18
 description @_@_TWAREN-TC-ASR9912-02_GigabitEthernet0/9/0/18_OLA-2
 no shutdown
interface GigabitEthernet0/9/0/18.1 l2transport
 description @_@_TWAREN-TC-ASR9912-02_GigabitEthernet0/9/0/18_OLA-2
 encapsulation untagged
l2vpn
 bridge group CUS
  bridge-domain OLA-2
   interface GigabitEthernet0/9/0/18.1
   vfi OLA-2
    vpn-id 753
    autodiscovery bgp
     rd 7539:vpn-id 753
     route-target 7539:vpn-id 753
     signaling-protocol ldp
```

送出 <=按下即送出並於設備commit生效!!

圖7.確認自動化頁面

4.2 介面狀態異常偵測

TWAREN 網管有蒐集路由器等設備日誌的規範，也因如此，將日誌轉送大資料庫分析成為必然的工作，日誌分析用意在於可透過日誌上的描述異常資訊，來判定是否有介面震盪(Flapping)的情況發生，震盪對於維運上造成困擾，例如設備介面發生震盪會造成 IGP 不斷收到網路拓樸變化的情形，導致路由器以重新學習方式計算最佳路徑，影響其網路，甚至於路由器不斷處理介面狀態更新，影響到 IGP 與 BGP 收斂，造成拉高 CPU 使用率，造成路由器全面性的影響。我們透過大資料庫(ELK)分析日誌，並且針對震盪發出電子郵件的告警。但為了確實檢查設備介面震盪造成對網路的衝擊，以自動化程式介入，評估震盪成的影響。下圖8為震盪發出的告警信，信件內指出發生在哪一台設備的哪一個介面。

Alert_time	TWAREN-TN-ASR9006-01	SG-MX240
2024/06/25 05:45	Error in receiving key exchange packet Interface TenGigE0/1/1/6 (#22TN_SG_10G), Flapping	EVENT <UpDown> xe-2/2/1 index 179 <Up Broadcast Multicast> address #0 4.69.8f.88.20.55 EVENT <UpDown> xe-2/2/1.0 index 356 211.79.48.114/30 -> 211.79.48.115 <Up Broadcast Multicast>

圖8 介面震盪告警信

也因此可以透過呼叫自動化檢查設備，我們檢查的步驟有，1.檢查設備介面、 2.檢查 OSPF 狀態、3.檢查 BGP 狀態。因此依序透過 Curl 執行以上三個步驟的自動化程式並獲取結果。自動化之結果如下圖9、圖10

圖9 介面檢查與 IGP 狀態

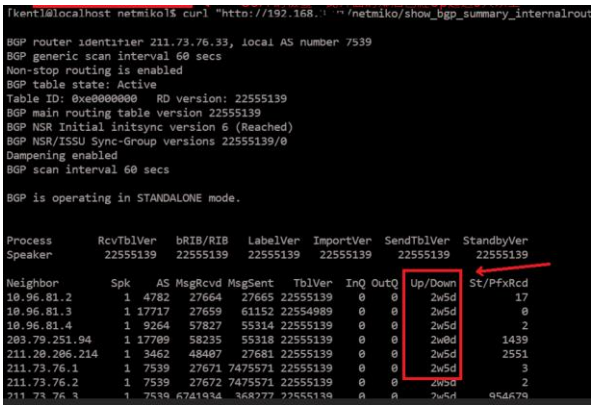

圖10 BGP 狀態

從圖9顯示出這兩組自動化程式行結果，分別為登入設備檢查介面是否為開啟(Up)，以及介面對 IGP 所造成的影響。第一組程式，如果介面顯示為 Down 肯定對網路有極大影響，但介面開啟也並不表示網路沒有受到影響，因此促成第二組程式的生成，透過『show ospf 7539 neighbor TenGigE0/1/1/6』指令，查詢介面狀態造成對鄰居(Neighbor)的影響，發現鄰居狀態已經有三天未變動，而不是在最近，即表示介面並未對路由造成傷害，如果是幾分鐘內的狀態改變，肯定對路由造成異動。最後，圖10是檢查是否對 BGP 造成影響，以『show bgp neighbor』指令，檢查鄰居狀態，也是圖中下方互連 IP(Peering IP)的狀態，如果是最近有異動，在 UP/Down 的時間欄位就會顯示幾分鐘，表示路由在幾分鐘前異動過，如果已經持續一段時間，如範例中的2w5d(兩週五天)即表

示路由正常，無異動，介面震盪對 BGP 不具影響。針對圖10列出太多持續時間很長的對連IP(Peering IP)狀態，其中我們不需要這些穩定的BGP 檢查資訊，透過後續程式處理，可以只列出持續時間小於30分鐘的資訊，這對於後續檢查，更容易判斷與造就自動化程式的人性化。

5. 結論

開發自動化程式對於網路上維運極為重要，透過程式語言的模組，撰寫除錯的腳本，猶如工程師登入設備下達指令，相信不僅僅是網路路由器可以派上用場，對於其他諸如防火牆、SDN、DDoS 防禦部屬，都可以找到它的切入點[5][6]，達到減少人為出錯與耗工時、安全的優點。

自動化不僅僅是減少人為的介入，也可以將其與其他系統連結，本篇論文連結介面異常偵測系統，以驗證介面變化所造成之衝擊，相同道理，例如在智能維運系統(AIOps)也串連自動化來達到智能維運的最後一哩路，相信在人工智慧不斷的受到重視的今天，必定佔有一席地位。

除了骨幹網路設備上的維運需求，內部網路上或資訊管理系統也可藉由自動化導入，達到事半功倍之效，近來防火牆廠商不斷提供不同程式語言 API [7]來看，自動化模組與 API 或許不再是彼此互斥之狀態，亦可使兩者相輔相成，達成維運之便利性。

參考文獻

[1] 陳俊傑“網際網路交換中心路由安全實作探討 -以 FOX 交換中心為例”,TANet 2023 Dec 2023.

[2] Kirk Byers, https://github.com/ktbyers/netmiko

[3] NTC-templates,https://github.com/networktocode/ntc-templates

[4] CURL, https://curl.se/

[5] 文/Jeff Vance, 譯/曾祥信,” 網路自動化最新趨勢”,CIO IT,2020, https://www.cc.ntu.edu.tw/chinese/epaper/0055/i114-P69-71-%E7%B6%B2%E8%B7%AF%E8%87%AA%E5%8B%95%E5%8C%96%E6%9C%80%E6%96%B0%E8%B6%A8%E5%8B%A2.pdf

[6] Kentik,https://www.kentik.com/

[7] Developer Docs ,https://pan.dev/panos/docs/