Interface-Statistiken in Cacti für Pirelli Router erstellen

In diesem Beitrag geht es darum, Cacti Interface-Statistiken auf einem Pirelli Router über den Umweg via SSH zu erzeugen. Dies wird auf bestimmten, kastrierten Pirelli-Geräten, wie sie von bestimmten Providern ihren Kunden zur Verfügung gestellt werden, auch dringend benötigt, da die Geräte den nötigen SNMP-Support schmerzlich vermissen lassen. Das fehlende SNMP Support lässt sich auf diesen Geräten auch mit Hilfe eines Hacks, der bereits seit mehreren Jahren in einschlägigen Foren kursiert, nicht nachrüsten.

Wer einen solcherart kastrierten Pirelli-Router sein Eigen nennt, dürfte daher an dieser Anleitung interessiert sein.
Zugegebenermassen muss man zwar mit der Kirche ums Dorf, aber es geht.
Der Trick liegt im Command Line Interface, welches per Telnet und SSH erreichbar ist. Hier kann mit dem Befehl "net ifcnt all" für jede Schnittstelle des Routers eine Statistik ausgegeben werden.
Ersetzt man das Schlüsselwort "all" durch die Interface-Bezeichnung, bspw. "wl0", erhält man die jeweiligen Einzelstatistiken.

Im folgenden Beispiel werden die Statistiken zur WAN-Schnittstelle angezeigt:

```
[~] # ssh admin@192.0.2.1
OpenRG> net ifcnt eth5.xx
---- Driver statistics: port "LAN Fiber VLAN xx" ----
 Device name: eth5.10 - Type: 48, Ethernet
 Network = WAN
 Port status = Connected
 Counters
   Rx Packets:       2006090
   Tx Packets:       2477100
   Rx Bytes:         835058499
   Tx Bytes:         2356068887
   Rx Pkts Errors:   0
   Tx Pkts Problems: 0
   Rx Dropped Pkts:  0
   Tx Dropped Pkts:  0
   Rx Multicast Pkts: NA
   Tx Multicast Pkts: NA
   Rx Broadcast Pkts: NA
   Tx Broadcast Pkts: NA
   Collisions:       NA

Returned 0
OpenRG>
```

Wer sich in der Pirelli Console schon umgesehen hat, ist vielleicht angesichts der vielen Schnittstellen etwas überfordert.
Die gute Nachricht: Die meisten davon braucht's nur für interne Zwecke und sind für die Überwachung kaum relevant. Daher hier eine Auflistung der notwendigen Schnittstellen und deren Funktion:

eth0 Switchport 1

eth1 Switchport 2
eth2 Switchport 3
eth3 Switchport 4
wl0 WLAN Access Point
eth5.xx WAN Schnittstelle (Fiber oder xDSL)

Damit diese Daten nun für Cacti nutzbar werden, bedarf es eines Hilfsprogramms, welches folgende Schritte durchführt:

- Login auf den Router
- Abrufen der Interface-Ststistiken
- Ausgabe der bereinigten Informationen

Gerade letzterem kommt eine grosse Bedeutung zu, da die Ausgabe wiederum scriptbasiert verarbeitet werden soll und somit normalisiert werden muss.
Wie dies im Detail aussieht, beschreibt die Cacti-Dokumentation.

Basierend auf einem Beispiel-Script aus dem py-expect Package habe ich eine stark abgewandelte Version erstellt, welche allein dem Auslesen des Pirelli dient und vollständig parametrisierbar ist.
Damit das Script ausführbar wird, sind zum Beispiel unter Debian die Packages python2.5 und python-pexpect erforderlich.

Das ganze sieht beim Ausführen dann wie folgt aus:

```
[~] # /opt/sbin/cg_ifstats.py -u admin -p password -h 192.0.2.1 -i wl0
RxBytes:4230422745 TxBytes:3249795610
```

Damit steht bereits der erste Baustein bereit. Nun muss Cacti noch entsprechend eingerichtet werden, damit die Script-Ausgabe zur Erstellung von Statistiken verwendet werden.

Anweisungen dazu finden sich ebenfalls in der Cacti-Dokumentation.

Die nachfolgenden Schritte verdeutlichen dies anhand einiger Screenshoots.
Man kann - und darf ;-) - es sich aber auch ein bisschen einfacher machen und einfach das Data Template mit den Abhängigkeiten direkt importieren.
Das Data Template kann zusammen mit dem Python Script hier heruntergeladen werden.

Wer die erforderlichen Schritte selbst durchführen möchte, erstellt zuerst eine sogenannte Date Input Method nach dem gezeigten Vorbild.
Dabei müssen insbesonders die Input wie auch die Output Fields erfasst werden (auf Schreibweise achten).
Die Data Input Method wird unter der Bezeichnung "Pirelli - IFSTATS" gespeichert.

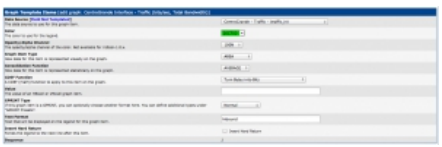
Anschliessend wird ein neues Data Template mit der Bezeichnung "Pirelli - Traffic" erzeugt.
Unter Data Input Method wird die zuvor erstellte Datenquelle "Pirelli - IFSTATS" verwendet.

Weiterhin benötigt das Data Template die Quelldatensätze, "traffic_in" und "traffic_out". Diese werden mit den einzelnen Zählern der Data Input Method "Pirelli - IFSTATS" verbunden, also "traffic_in" mit "RxBytes" und "traffic_out" mit "TxBytes".
Wichtig ist, dass als "Data Source Type" der Typ "COUNTER" verwendet wird.

Unter "Custom Data" wird bei allen Feldern die Checkbox "User Per-Data Source Value" markiert.
Dies ist erforderlich, damit beim folgenden Erzeugen der Graphen die Werte pro Host und Datenquelle einzeln definiert werden können.

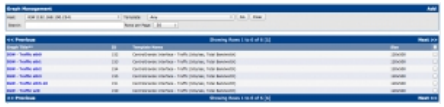
Nun fehlt das Graph Template. Hierfür klont man am einfachsten eines der bestehenden Graph Templates, beispielsweise "Interface - Traffic (bits/sec, Total Bandwidth)".
Das geklonte Graph Template bedarf noch einiger Anpassungen, insbesonders bei den Graph Template Items und Graph Item Inputs.
Als Letztere dienen die bereits zuvor definierten Datenquellen "traffic_in" und "traffic_out".

Die Datenquell muss auch bei den Graph Template Items noch entsprechend zu "traffic_in" und "traffic_out" angepasst werden.



Damit wäre soweit alles vorbereit, dass nun ein Host-Objekt für den Pirelli erfasst werden kann.
Anschliessend werden dem Host-Objekt neue Graphs hinzugefügt.

Je nach Bedarf werden für einzelne oder alle Schnittstelle (wl0, eth5.10, eth0, usw) die Graphen erzeugt.
Dazu muss man zur besseren Unterscheidung unter Title auch die jeweilige Schnittstellenbezeichnung ergänzt werden.
Ferner sind in den Feldern für Username, Passwort, Hostname und Interface die erforderlichen Daten einzutragen.



Nach Abschluss aller Schritte lassen sich fortan die erzeugten Graphs in Cacti bewundern.