RIP-Laborübung

IN ARBEIT

Hinweis-Seite:

Offizielle Dokumentation zu quagga: http://www.nongnu.org/quagga/docs.html

HowTo: http://opentodo.net/2012/08/configuring-routing-protocols-with-quagga/

Allgemeine Informationen zu RIP

Unter Linux kann das Software-Paket quagga verwendet werden, um die einzelnen Routing Protokolle zu steuern. Folgende Module stehen unter quagga zur Verfügung:

Protokoll	Modulname in quagga
Konfiguration-Frontend für quagga	`zebra`[1]
RIPv1/RIPv2 für IPv4	`ripd`[2]
RIPv1/RIPv2 für IPv6	`ripngd`[3]
OSPFv2/OSPFv3 für IPv4	`ospfd`[4]
OSPFv2/OSPFv3 für IPv6	`ospf6d`[5]
BGP für IPv4/IPv6	`bgpd`[6]
IS-IS für IPv4/IPv6	`isisd`[7]

Daneben gibt es noch die Module `babeld`, `osrd`, `ldpd` und `bfdd` auf die hier nicht weiter eingegangen wird.

Die gesamte Architektur sieht wie folgt aus:

BILD mit quagga-Modulen

Der `zebra`-Daemon[8] stellt als Routing-Manager die übergeordnete Schnittstelle zu den Untermodulen dar. Das `zebra`-Modul stößt u.a. die Aktualisierung der Kernel-Routing-Table Einträge an, beobachten die Schnittstellen und tauschen Routing-Informationen zwischen den Routing Modulen aus.

Routing aktivieren (FORWARDING)

Um einen PC zum Router zu machen, muss dieser in den `FORWARDING`-Modus versetzt werden. Das heißt der PC kann Datenpakete über Schnittstellen hinweg weiterleiten. Normalerweise behandelt ein PC jede Schnittstelle getrennt und würde die Datenpakete nicht weiterleiten.

Unter Linux geschieht dies über den folgenden Befehl:

```
sudo sysctl net.ipv4.ip_forward=1
```

Über den folgenden Befehl kann der aktuelle Status des FORWARDING überprüft werden:

```
cat /proc/sys/net/ipv4/ip_forward
1: FORWARDING aktiviert
0: FORWARDING deaktiviert
```

Einrichten der virtuellen Maschine

Es muss als Router eine VM genutzt werden, die mindestens zwei Schnittstellen besitzt.

Quagga einrichten

Der quagga-Daemon speichert seine Konfiguration unter /etc/quagga/. Dort liegen nach der Installation zu nächst nur zwei Dateien:

Dateiname	**Bedeutung**
/etc/quagga/daemons	Hier werden die einzelnen Module mit yes/no aktiviert/deaktiviert
/etc/quagga/debian.conf	Hier werden die Ports und IP-Adressen der einzelne Konfigurationsschnittstellen gesetzt.

Damit überhaupt ein Routing Protokoll gestartet wird, muss das entsprechende Modul in der /etc/quagga/daemons auf yes gesetzt werden.

```
zebra=yes
...
ripd=yes
...
```

Damit die einzelnen Module wissen, was zu tun ist, benötigen auch diese eine Konfigurationsdatei. Am besten nutzt mal als Ausgangspunkt die mitgelieferten Beispielkonfigurationen. Diese liegen im folgenden Ordner:

```
/usr/share/doc/quagga/examples/
```

Mit dem folgenden Befehl, werden die beiden Beispielkonfigurationen für zebra und ripd in den /etc/quagga/-Ordner kopiert.

```
sudo cp /etc/share/doc/quagga/examples/zebra.conf.sample
/etc/quagga/zebra.conf
sudo cp /etc/share/doc/quagga/examples/ripd.conf.sample
/etc/quagga/ripd.conf
```

HINWEIS: Wichtig ist, dass die Dateien im /etc/quagga/-Ordner korrekt geschrieben werden, da sie ansonsten nicht gefunden werden.

Nun kann der quagga-Daemon gestartet werden, so dass dieser die gewünschten Module startet.

```
sudo /etc/init.d/quagga [start/restart/stop]
```

HINWEIS: In den eckigen Klammern werden alle möglichen Commands an den quagga aufgeführt. Es kann jeweils nur EINER verwendet werden.

Die folgende Ausgabe zeigt, dass die Konfiguration von zebra offensichtlich fehlt (`not started without config file`), aber die Konfiguration von `ripd` (`ripd` sonst nichts) vorhanden ist.

```
Loading capability module if not yet done.
Starting Quagga daemons (prio:10): zebra (not started without config file)
ripd.
Starting Quagga monitor daemon: watchquagga.
```

Um zu überprüfen, ob der Daemon korrekt läuft kann man sich den entsprechenden Prozess anzeigen lassen:

```
ps aux | grep quagga
```

HINWEIS: `ps aux` listet alle Prozesse inklusiver der Auslastung auf. Mit `|` wird diese Ausgabe an den `grep`-Befehl weitergeleitet, der wiederum in der Ausgabe den String `quagga` sucht.

Die Ausgabe könnte so aussehen:

```
ps aux | grep quagga

quagga    5496  0.0  0.0  24288   980 ?        Ss   17:30   0:00
/usr/lib/quagga/ripd --daemon -A 127.0.0.1
root      5501  0.0  0.0  15364   508 ?        Ss   17:30   0:00
/usr/lib/quagga/watchquagga --daemon zebra ripd
root      5503  0.0  0.0  16660   904 pts/6    S+   17:35   0:00 grep --
color=auto quagga
```

Konfiguration des zebra-Moduls

Man muss sich die einzelnen Module wie einen „Standalone“-Router vorstellen, der per Remote-Zugriff konfiguriert wird. Da man direkten Zugriff auf die Maschine selbst hat, ist die `remote`-Adresse der eigene Rechner also `localhost`:

```
telnet localhost 2601
```

AUSGABE:

```
Hello, this is Quagga (version 0.99.22.1).
Copyright 1996-2005 Kunihiro Ishiguro, et al.


User Access Verification

Password:
```

Als Passwort verwendet man `zebra` bzw. das Passwort, das man in der `zebra`-Konfigurationsdatei (`/etc/quagga/zerbra.conf`[9]) gesetzt hat.

Die folgende Tabelle zeigt die wichtigsten Befehle des `zebra`-Moduls. Mit `?` bzw. `list` kann jederzeit die Hilfe angezeigt werden (vgl. STP-Übung).

Befehl	**Funktion**	**Beispiel**
`show`	Zeigt die aktuelle Konfiguration an z.B. die Routing-Tabelle	`show ip route`
`enable/disable`	Aktiviert/deaktiviert den Router	`enable` oder `disable`
`configure terminal`	Wechseln in den Konfigurationsmodus. Neues Prompt:`Router(config)`	`configure terminal`
`interface`	Auswählen der Schnittstelle, die konfiguriert werden soll. Im Prompt:`Router(config)`	`interface eth0`
`ip/no ip`	Setzen/Löschen einer IP-Adresse für das ausgewählte Interface. Im Prompt: `Router(config-if):`	`ip address 10.0.0.1/8`
`shutdown/no shutdown`	Schnittstelle aktivieren/deaktivieren. Im Prompt: `Router(config-if):`	`no shutdown`
`hostname`	Setzen des Router-Namens.	`hostname <NEUERNAME>`
`write`	Dauerhaftes Speichern der aktuellen (`running-config`). Ansonsten wird nach dem nächsten Restart alles zurückgesetzt.	`write`

Beispiel Befehlsfolge im CLI[10]:

```
telnet localhost 2601
Password: zerbra

Router>enable
Password: zebra
Router#configure terminal
Router(config)#interface eth0
Router(config-if)#ip address 10.0.0.1/8
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#hostname tralala
tralala(config-if)#show running-config

Current configuration:
!
hostname tralala
password zebra
enable password zebra
!
interface eth0
 ip address 10.0.0.1/8
 ipv6 nd suppress-ra
!
interface eth1
```

```
 ipv6 nd suppress-ra
!
interface lo
!
!
!
line vty
!
end
```

Alternativ zur direkten Eingabe der Kommandos, kann auch eine Textdatei als Konfigurationsdatei herangezogen werden. Der Befehlt `show running-config` gibt die aktuelle Konfiguration aus, die für die Textdatei genutzt werden kann. Für das obige Beispiel würde demnach folgender Inhalt darin stehen:

zerbra.conf_sample

```
!
hostname tralala
password zebra
enable password zebra
!
interface eth0
 ip address 10.0.0.1/8
 ipv6 nd suppress-ra
!
interface eth1
 ipv6 nd suppress-ra
!
interface lo
!
!
!
line vty
!
end
```

Konfiguration des ripd-Moduls

```
telnet localhost 2601
```

1)

Kernel-Interface, Statische Routen

2)

`ripd`: **R**outing **I**nformation **P**rotocol **D**aemon

3)

`ripngd`: **R**outing **I**nformation **P**rotocol **N**ext **G**eneration **D**aemon

4)

`osfpd`: **O**pen **S**hortest **P**ath **F**irst **D**aemon

5)

`osfp6d`: **O**pen **S**hortest **P**ath **F**irst IPv**6** **D**aemon

6)

`bgpd`: **B**order **G**ateway **P**rotocol **D**aemon

7)

`isisd`: **I**ntermediate **S**ystem to **I**ntermediate **S**ystem **D**aemon

8)

Daemon: Ein Daemon stellt unter Linux einen Dienst bereit. Z.B. Der Druckerwarteschlagen-Daemon `cups`

9)

hier kann auch der Name des Routers gesetzt werden

10)

CLI: **c**ommand **l**ine **i**nterface; Kommandozeile des `zebra`/`ripd`-Daemons