Linux Networking

Markus Gachnang und Martin Sprecher

1. Oktober 2020

Inhaltsverzeichnis

1 Ausgangslage

2 Kochbuch

It is expected of you to hand in a step-by-step cookbook for the whole final setup. Explain important commands and reason your decisions. We should be able to fully retrace what you did to be able to assess your work. One cookbook is expected per group.

2.1 General

Change the password ... Also, change the hostname to the name given in LTB.

Wir verbinden uns auf jeden Container und ändern den Inhalt der Datei `/etc/hostname` auf den Namen des Containers. Dafür benützen wir `sudo nano /etc/hostname`, ändern den Namen und speichern mit Ctrl-O und beenden nano mit Ctrl-X. Zusätzlich rufen wir `sudo hostname <newHostName>` auf.
Wird setzten das Passwort des jeweiligen Containers auf seinen Namen mit `sudo passwd ins`.

2.2 IP Address Assignment

Use Netplan to assign the ip addresses to the interfaces. ...

Name	**IP**
Client	ENS2: 172.16.0.2
R1	ENS2: 172.16.0.1 ENS3: 10.0.1.1
R2	ENS2: 10.0.1.2 ENS3: 10.0.4.1 ENS4: 10.0.2.1
R3	ENS2: 10.0.2.2 ENS3: 10.0.5.1 ENS4: 10.0.3.1
R4	ENS2: 10.0.4.2 ENS3: 10.0.5.2 ENS4: 10.0.100.1
R5	ENS2: 10.0.3.2 ENS3: 192.168.1.1
Server	?: 192.168.1.100
MITM	ENS2: 10.0.100.2

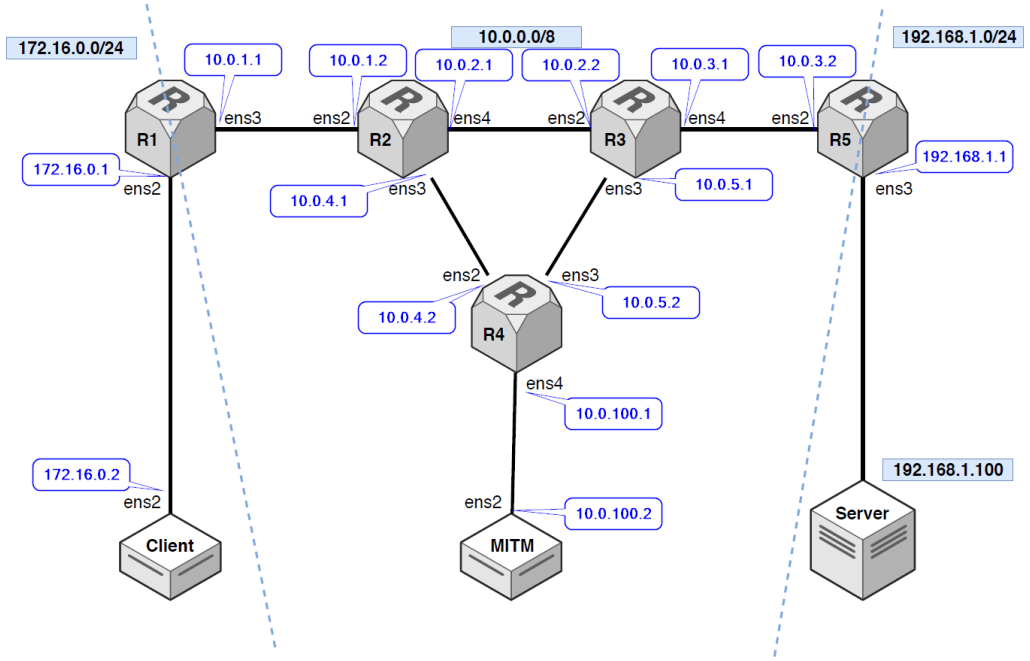

Abbildung 1: Netplan

Die Adressen sind so definiert, dass jede Verbindung ein eigenes Subnet hat. Bird sucht seine Nachbarn anhand des Broadcasts.
Wir ändern den netplan mit `sudo nano /etc/netplan/50-cloud-init.yaml`.

```
# This file is generated from information provided by
# the datasource.  Changes to it will not persist across an instance.
# To disable cloud-init's network configuration capabilities, write a file
# /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg with the following:
# network: {config: disabled}
network:
    version: 2
    ethernets:
        ens2:
            dhcp4: false
            addresses: [172.16.0.1/24]
        ens3:
          dhcp4: false
          addresses: [10.0.1.1/24]
```

Listing 1: /etc/netplan/50-cloud-init.yaml on R1

Mit `netplan apply` übernehmen wir die neu definierten IP-Adressen.

2.3 BIRD

Now, your routers must run OSPF. ...

- OSPFv2 must run on the routers.
- Find a way to protect the CPU from too much OSPF processing.
- A restart of BIRD should not result in lost routes.

Wir definieren die Configuration von Bird indem wir "/etc/bird/bird.conf" anpassen.
Um die Auslastung von CPU und Netzwerk zu vermindern, haben wir uns an [1] orientiert.

```
# This file is generated from information provided by $0
# Please refer to the documentation in the bird-doc package or BIRD User's
```

```
3  # Guide on http://bird.network.cz/ for more information on configuring BIRD
       and
4  # adding routing protocols.
5
6  # log "/var/log/bird.log" all;
7  log syslog { info, remote, warning, error, auth, fatal, bug };
8
9  # Change this into your BIRD router ID.
10 router id 1.1.1.1;
11
12 # The Device protocol is not a real routing protocol. It doesn't generate
       any
13 # routes and it only serves as a module for getting information about
       network
14 # interfaces from the kernel.
15 protocol device {
16     scan time 10; # Scan interfaces every 10 seconds
17 }
18
19 # The Kernel protocol is not a real routing protocol. Instead of
       communicating
20 # with other routers in the network, it performs synchronization of BIRD's
21 # routing tables with the OS kernel.
22 protocol kernel {
23     metric 64;      # Use explicit kernel route metric to avoid collisions
24                     # with non-BIRD routes in the kernel routing table
25     persist;        # Don't remove routes on BIRD shutdown
26     scan time 20;   # Scan kernel routing table every 20 seconds
27     import none;
28     export all;     # Actually insert routes into the kernel routing table
29 }
30
31
32 protocol rip {
33     export all;
34     import all;
35     interface "*";
36 }
37
38 protocol static {
39         import all;
40
41         172.16.0.0/24 via "ens2"
42 }
43 protocol ospf {
44     tick 10;       # The routing table calculation and clean-up of areas'
           databases is not performed when a single link
45                     # state change arrives. To lower the CPU utilization,
                           it's processed later at periodical intervals of num
46                     # seconds. The default value is 1.
47     import all;
48     #export filter {
49     #         ospf_metric1 = 1000;
50     #         if source = RTS_STATIC then accept; else reject;
51     #};
52
53     area 0 {
54         networks {
55             10.0.0.0/8;
56             172.16.0.0/24;
57             192.168.1.0/24;
58         };
59
60         interface "ens*" {
61             cost 5;
62             type broadcast;
```

```
63              hello 5;
64              retransmit 2;
65              wait 10;
66              dead 20;
67          };
68
69          interface "*" {
70                  cost 1000;
71                  stub;
72          };
73      };
74  }
```

Listing 2: /etc/bird/bird.conf on R1

Zuerst definierten wir die "router id" nach Vorgabe. Unter "protocol kernen" definierten wird

dass alle routes in die kernel routing table exportiert werden und dort persistent sind, dadurch funktioniert das Routing weiterhin wenn Bird beendet wird.

Unter "protocol static" werden statische routes eingetragen. Nur R1 und R5 haben dort Einträge um ein routing nach Aussen / Innen des Routing-Netzwerkes zu ermöglichen.

Unter "protocol ospf" definieren wird das Verhalten des OSPF. Wir setzten "tick" auf 5, dadurch werden Änderungen innerhalb von 5 Sekunden gesammelt, bevor diese ausgeführt werden. Dadurch wird das Netzwerk weniger geflutet und die CPU ausgelastet.
Wir definieren dass die "area 0" aus den Netzwerken "10.0.0.0/8", "172.16.0/24" und "192.168.1.0/24" besteht. Dadurch weiss OSPF, welche Adressen geroutet werden sollen.
Zusätzlich definieren wir, welche Interface von OSPF verwendet werden sollen. Dort definieren wir auch, in welchem Intervall zum Beispiel ein "hello" geschickt wird. Um ebenfalls das fluten des Netzwerks zu minimieren, werden gewisse Werte höher gesetzt als der Standartwert.
Zum Schluss definieren wir, das alle restlichen Interfaces "stub" sind und daher zu ignorieren sind.

Wir haben eine Vorlage aus dem Internet übernommen [2] und die einzelnen Einstellungen anhand von [3] überprüft und angepasst.

2.4 IP Forwarding

"IP-Forwarding" ist standartgemäs ausgeschaltet. Mit `sysctl net.ipv4.ip_forward=1` aktivieren wird das weiterleiten der Packete erlaubt.

2.5 Script

Da wir keine Ahnung von "Bird" hatten, wurde das Anpassen und Testen der Configuration aufwendig, weshalb wir kurzer Hand ein Script erzeugten, welche die zuvor definierten Einstellungen anwendet:

```
1  #!/bin/bash
2
3  setup_hostname()
4  {
5      echo "Set hostname to $1"
6      echo "$1" > '/etc/hostname'
7      hostname $1
8  }
9
10 setup_ip()
11 {
12     echo "Setup netplan"
13     cat <<EOF > '/etc/netplan/50-cloud-init.yaml'
14 # This file is generated from information provided by $0
15 # Changes to it will not persist across an instance.
```

```
# To disable cloud-init's network configuration capabilities, write a file
# /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg with the following:
# network: {config: disabled}
network:
    version: 2
    ethernets:
EOF
    if [ $# -ge "1" ]; then
        ifconfig ens2 down
        echo " ens2 = $1"
        cat <<EOF >> '/etc/netplan/50-cloud-init.yaml'
        ens2:
            dhcp4: false
            addresses: [$1]
EOF
    fi
    if [ $# -ge "2" ]; then
        ifconfig ens3 down
        echo " ens3 = $2"
        cat <<EOF >> '/etc/netplan/50-cloud-init.yaml'
        ens3:
            dhcp4: false
            addresses: [$2]
EOF
    fi
    if [ $# -ge "3" ]; then
        ifconfig ens4 down
        echo " ens4 = $3"
        cat <<EOF >> '/etc/netplan/50-cloud-init.yaml'
        ens4:
            dhcp4: false
            addresses: [$3]
EOF
    fi

    netplan apply

    if [ $# -ge "1" ]; then
        ifconfig ens2 up
    fi
    if [ $# -ge "2" ]; then
        ifconfig ens3 up
    fi
    if [ $# -ge "3" ]; then
        ifconfig ens4 up
    fi
}

setup_bird()
{
    echo "Setup bird"
    cat <<EOF > '/etc/bird/bird.conf'
# This file is generated from information provided by $0
# Please refer to the documentation in the bird-doc package or BIRD User's
# Guide on http://bird.network.cz/ for more information on configuring BIRD
    and
# adding routing protocols.

# log "/var/log/bird.log" all; # Log all in logfile: Commented out because
   af access error..
log syslog { info, remote, warning, error, auth, fatal, bug };

# Change this into your BIRD router ID.
router id $1;

```

```
79  # The Device protocol is not a real routing protocol. It doesn't generate
        any
80  # routes and it only serves as a module for getting information about
        network
81  # interfaces from the kernel.
82  protocol device {
83      scan time 10; # Scan interfaces every 10 seconds
84  }
85
86  # The Kernel protocol is not a real routing protocol. Instead of
        communicating
87  # with other routers in the network, it performs synchronization of BIRD's
88  # routing tables with the OS kernel.
89  protocol kernel {
90      metric 64;      # Use explicit kernel route metric to avoid collisions
91                      # with non-BIRD routes in the kernel routing table
92      persist;        # Don't remove routes on BIRD shutdown
93      scan time 20;   # Scan kernel routing table every 20 seconds
94  #    import none;    # Default is import all
95      export all;     # Actually insert routes into the kernel routing table
96  }
97
98
99  protocol rip {
100     export all;
101     import all;
102     interface "*";
103 }
104
105 protocol static {
106         import all;
107
108 EOF
109
110     for var in "$@"
111     do
112         echo " $var"
113         if [ "$var" != "$1" ] && [ "$var" != "$2" ]; then
114             echo "        route $var;" >> '/etc/bird/bird.conf'
115         fi
116     done
117
118     cat <<EOF >> '/etc/bird/bird.conf'
119 }
120
121 protocol ospf {
122     tick 5;       # The routing table calculation and clean-up of areas'
            databases is not performed when a single link
123                   # state change arrives. To lower the CPU utilization,
                          it's processed later at periodical intervals of num
124                   # seconds. The default value is 1.
125     import all;
126     #export filter {
127     #        ospf_metric1 = 1000;
128     #        if source = RTS_STATIC then accept; else reject;
129     #};
130
131     area 0 {
132         networks {
133             10.0.0.0/8;
134             172.16.0.0/24;
135             192.168.1.0/24;
136         };
137
138         interface "$2" {
139             cost 10;
```

```
140             type broadcast;
141             hello 9;
142             retransmit 6;
143             wait 50;
144             dead 5;
145         };
146
147         interface "*" {
148                 cost 1000;
149                 stub;
150         };
151     };
152 }
153
154 EOF
155     sysctl net.ipv4.ip_forward=1
156     ip route flush table main
157     bird -p
158     birdc down
159     # bird -R
160     systemctl start bird
161     birdc show status
162     systemctl status bird
163 }
164
165 setup()
166 {
167     echo "Start setup for '$1'"
168     case $1 in
169         Client)
170             setup_hostname "Client"
171             setup_ip "172.16.0.2/24"
172             echo "            gateway4: 172.16.0.1" >> '/etc/netplan/50-
                    cloud-init.yaml'
173             ;;
174         MITM)
175             setup_hostname "MITM"
176             setup_ip "10.0.100.2/24"
177             echo "            gateway4: 10.0.100.1" >> '/etc/netplan/50-
                    cloud-init.yaml'
178             ;;
179         R1)
180             setup_hostname "R1"
181             setup_ip "172.16.0.1/24" "10.0.1.1/24"
182             setup_bird "1.1.1.1" "ens3" "172.16.0.0/24 via \"ens2\""
183             # "10.0.0.0/8 via \"ens3\""
184             ;;
185         R2)
186             setup_hostname "R2"
187             setup_ip "10.0.1.2/24" "10.0.4.1/24" "10.0.2.1/24"
188             setup_bird "2.2.2.2" "ens*"
189             # "10.0.1.0/24 via \"ens2\"" "10.0.4.0/24 via \"ens3\""
                    "10.0.3.0/24 via \"ens4\""
190             ;;
191         R3)
192             setup_hostname "R3"
193             setup_ip "10.0.2.2/24" "10.0.5.1/24" "10.0.3.1/24"
194             setup_bird "3.3.3.3" "ens*"
195             #"10.0.2.0/24 via \"ens2\"" "10.0.4.0/24 via \"ens3\""
                    "10.0.5.0/24 via \"ens4\""
196             ;;
197         R4)
198             setup_hostname "R4"
199             setup_ip "10.0.4.2/24" "10.0.5.2/24" "10.0.100.1/24"
200             setup_bird "4.4.4.4" "ens*"
201             ;;
```

```
202         R5)
203             setup_hostname "R5"
204             setup_ip "10.0.3.2/24" "192.168.1.1/24"
205             setup_bird "5.5.5.5" "ens2" "192.168.1.0/24 via \"ens3\""
206             # "10.0.0.0/8 via \"ens2\""
207             ;;
208         *)
209             echo "name is unknewn..."
210             exit 1
211             ;;
212     esac
213 }
214
215 echo "CldInf Networker"
216 if [ "$EUID" -ne 0 ]; then
217     echo "Please run as root"
218     exit 1
219   elif [ $# -lt "1" ]; then
220     hostname=$(hostname)
221     case $hostname in
222         Client|MITM|R1|R2|R3|R4|R5)
223             setup $hostname
224             ;;
225         *)
226             echo "Usage: $0 <name>"
227             echo " name = Client, MITM, [R1 .. R5]"
228             exit 1
229             ;;
230     esac
231   else
232     setup $1
233 fi
```

Listing 3: /doConfig.sh

3 Verifizierung

> Verify your routing implementation. Explain which exact commands you used for each verification step and show how they provide prove that your setup works.

Zuerst haben wir die IP-Vergabe überprüft und von jedem Router aus seine Nachbarn mit dem Befehl `ping` angepingt. So haben wir als Beispiel von "R3" aus "R2", "R4" und "R5" angepingt. Alle Pings waren erfolgreich.

Anschliessend haben wir von "R1" jedes andere Gerät mit `ping` angepingt und so erkennen können, dass die Routes funktionieren. Zusätzlich haben wir auch eine Routes genauer mit `tcptraceroute` untersucht und konnten die Verbindung anhand des Netplans (Abbildung 1) nachverfolgen.

3.1 Route Failover

> Verify that your setup works. Prove that a route failover takes place in case ofa route outage. To do that, a well known tool can be used.

Uns ist leider kein solches "well known tool" bekannt, weshalb wir manuell gewisse Interfaces der Router mit `ifconfig ens2 down` ausgeschaltet haben während ein anderer Routern `ping` ausgeführt hat. Es dauerte zwar einige Sekunden (etwa 20 Sekunden) bis die Verbindung wieder stand, aber der Route Failover funktionierte.

Wir haben von "R1" aus gepingt und bei "R2" jeweils die Interface aus- und eingeschaltet.

3.2 Passive Interfaces

Show that no OSPF packets are sent into the client and server networks,too.`tcpdump` and `tshark` are good tools for that, sniff on the suspicious interfaces and filter for OSPFv2 packets. To be sure that your filter works, sniff on an interface where you expect OSPF-messages, too.

Wir haben versucht, anhand von Wireshark mit "Sshdump" und "Ciscodump" auf den Interfaces zu lauschen, erhielten aber keine Pakete. Wir versuchten ebenfalls mit `tcpdump` die Pakete aufzufangen. Dies klappe zwar, aber das pcap-File wollte sich einfach nicht übertragen lassen. Wir sind anhand von [4] vorgegangen.

Wir können es zwar nicht beweisen, aber anhand von der Konfiguration von Bird und OSPF haben wir definiert, dass jeweils nur die Interfaces aktiv sind, bei welchem weitere Router sind. So hat "R1" nur ein Interface "ens3" auf welches Bird reagiert. Das gleiche ist bei "R5" bei welchem nur das Interface "ens2" konfiguriert ist. Alle anderen interfaces sind als "stub" definiert.

3.3 Access Website

Finally, you must be able to access the webpage from the webserver. You can use `curl` or `wget` for that. The webserver listens on port 8080.

4 Performanz

Provide the measurement before and after the appliance of the `tc` commands.
Provide the exact used `tc` commands.

5 Referenzblatt

We also expect you to hand in a reference sheet for all the net-work commands used in this lab. Just list every command and its function. This reference must not be longer than one page.

Befehl	Funktion
`sudo nano <Pfad>`	Öffnen eines Files im Texteditor
`sudo hostname < newHostName>`	Zum ändern des Hostnames
`sudo passwd`	Ändert das Passwort des aktuellen Benutzers.
`sudo birdc`	Zum kommunizieren mit einem laufenden BIRD. Einzelne Befehle, die hier nicht aufgeführt sind, sind auf [5] zu finden.
`sudo birdc show status`	Anzeigen vom router status, BIRD version, Laufzeit und Zeitpunkt von der letzten Rekonfiguration.
`sudo birdc show interfaces`	Anzeigen der Liste von allen Interfaces. Zeigt für jedes Interface, Typ, Status, MTU und die zugewiesene Adresse an.
`sudo birdc show ospf interface`	Anzeigen von detailierten Informationen über die OSPF Interfaces.
`birdc show ospf neighbors`	Anzeigen der Liste mit allen OSPF Nachbaren deren Zustand.
`birdc show ospf state`	Anzeigen von detailierten Informationen über OSPF Bereiche basierend auf der link-state database. Es zeigt die Netzwerk Topolgie, stub Netzwerke, zusammengeführte Netzwerke und Router von anderen Areas und externen Routen. Ausserdem zeigt es erreichbare Netzwerk Knoten an.
`ping -c <num> <ip>`	sendet num mal einen Ping an die ip
`tcptraceroute <ip>`	Zum Anzeigen vom Pfad im Netzwerk vom Host, auf dem die Traceroute ausgeführt wird, und der angegebenen IP, sowie dem Ort, an dem die Route, falls vorhanden, nicht abgeschlossen werden kann und allen Hops bis zum Ziel.
`sysctl net.ipv4. ip_forward=1`	Mit dem Befehl wird das weiterleiten der Packete erlaubt.
`sudo reboot`	Neustarten des Geräts.
`blub`	blob

6 Anhänge

6.1 Routing

- Your delivered report must include the new usernames and pass-words of the hosts.
- Your delivery must contain the created netplan files and an ad-dress plan.
- Attach the BIRD config files to your cookbook and explain how you achieve the minimal requirements.
- Verify your routing implementation. Explain which exact commands you used for each verification step and show how they provide prove that your setup works.

Erarbeitung der BIRD-config ist im 2.3 erläutert. Verifikation der Routing unter 3.

6.2 Firewall

The next step is to implement a firewall in your network. On Linux, netfilter was the kernel firewall for a long time. To configure netfilter, tools like iptables, ip6tables, arptables and so on were used. But for this task, you'll have to use the more modern nftables and its nft utility to configure it. First, check all open ports of the webserver from your client with nmap. Your firewall solution must fulfill the following requirements:

- You must use nftables and its utility nft. Do not use iptables.
- The firewall must run on one of the routers.
- The website is only accessible from the clients network (172.16.0.0/24).
- Pings must not be answered.
- nmap must not show any open ports. Check if your firewall works by executing nmap again.

We're interested in the used `nmap` command, where the firewall runs and why you've choosen that location. Print the ruleset of the firewall and attach it to your report.

6.3 Quellennachweis

[1] G. Huston, M. Rossi, and G. Armitage, *A Technique for Reducing BGP Update Announcements through Path Exploration Damping*. Selected Areas in Communications, October 2010. [Online]. Available: https://www.potaroo.net/papers/ieee/bgp_updates_2010.pdf

[2] Ondrej Sury, "Ospf_example · wiki · labs / bird internet routing daemon · gitlab," 25.09.2020. [Online]. Available: https://gitlab.nic.cz/labs/bird/-/wikis/OSPF_example#:~:text=OSPF%20example,ptp%20links%20to%20other%20routers.

[3] Tomas Filip - HIPPO © 2004 and C. F. ALL: WEBarium, "The bird internet routing daemon project," 25.09.2020. [Online]. Available: https://bird.network.cz/?get_doc&v=20&f=bird-6.html

[4] A. Phillips, "How to run a remote packet capture with wireshark and tcpdump," *Comparitech*, 27.12.2018. [Online]. Available: https://www.comparitech.com/net-admin/tcpdump-capture-wireshark/

[5] Tomas Filip - HIPPO © 2004 and C. F. ALL: WEBarium, "The bird internet routing daemon project," 25.09.2020. [Online]. Available: https://bird.network.cz/?get_doc&v=20&f=bird-4.html

7 Nachwort

Kein Mitglied unserer Gruppe hat CN2 besucht. Alleine bis wir 2.3 zum laufen brachten haben wir gemeinsam mehr als 30 Stunden aufgewendet.