Lösung 08: Routing, IPv4 Fragmentierung

Aufgabe 1: IPv4 Fragmentierung in Wireshark

a) Protokoll: ICMP 0x01
b) Header Length: 20 Byte, Total Length 84 Byte → Nutzlast: 64 Bytes
c) Nein, da das „More Fragments Bit“ 0 ist und da der Fragment Offset 0 ist.
d) Time-To-Live wird mit jedem ICMP Echo Request um 1 größer. Dieses Verhalten wird für Traceroute benötigt.
Ferner ändert sich auch jeweils die Header Checksum, da z.B. TTL in das Checksum Feld mit einberechnet wird.
e) Man sieht dies daran, dass das „More Fragment bit“ auf 1 gesetzt ist. Die Tatsache, dass „Fragment Offset“ gleich 0 ist, zeigt an, dass dies das 1. Fragment des Datagramms ist. Die Gesamtlänge des ersten Fragments ist 1500 Byte (1480 Bytes Nutzlast und 20 Byte IP Header).
f) Paket Nr. 93: Fragment Offset ist 1480, von daher kann es nicht das 1. Fragment sein. Es handelt sich um das letzte Fragment, da „More Fragments“ gleich 0 ist.
Hinweis: Wireshark markiert das 1. Fragment als „Fragment“, das 2. Fragment ist nicht so klar erkennbar, da es als „Echo (ping) request“ markiert wird. Grundsätzlich ermittelt Wireshark immer für das letzte Fragment die dahinterliegende Anwendung bzw. die darüberliegenden Protokolle.

Aufgabe 2: Hierarchisches Routing

a) Neben dem Prefix ist der **AS-Pfad** und der **Next-Hop** enthalten. Der AS-Pfad enthält alle ASe in der Reihenfolge vom aktuellen Netz bis zum Zielnetz. Die Länge des AS-Pfades kann bei der Auswahl einer besten Route entscheidend sein. Ferner könnte man mittels des AS-Pfades „Schleifen“ im Routing erkennen. Der Next-Hop gibt an, bei welchem Router die Inter-AS-Route beginnt. Der Weg dorthin kann dann mittels eines Intradomain Routingprotokolls bestimmt werden.

b) Es gibt hier nur einen Interdomain-Pfad nämlich: AS3-AS4. Deshalb lernt Router 1d als Beginn des Interdomain-Pfades den BGP Next-Hop 3a kennen und muss einen Weg zu diesem Interdomain Pfad über ein Intradomain Routingprotokoll (im konkreten Fall RIP) bestimmen. Als Interface wird I_1 bestimmt, da der Weg zum Next-Hop 3a direkt über 1a am kürzesten ist, also z.B. Beispiel kürzer als 1b-1a-1c.

c) Im konkreten Fall haben beide AS-Pfade die gleiche Länge. Die Gateways bzw. BGP Next-Hops sind 3a bzw. 2a. Da der Next-Hop 2a von 1d schneller erreichbar ist als der Next-Hop 3a, wird nun als Ausgangs-Interface I_2 gewählt.

d) Jetzt wird I_1 gewählt, da zunächst immer geschaut wird, welcher AS-Pfad kürzer ist.

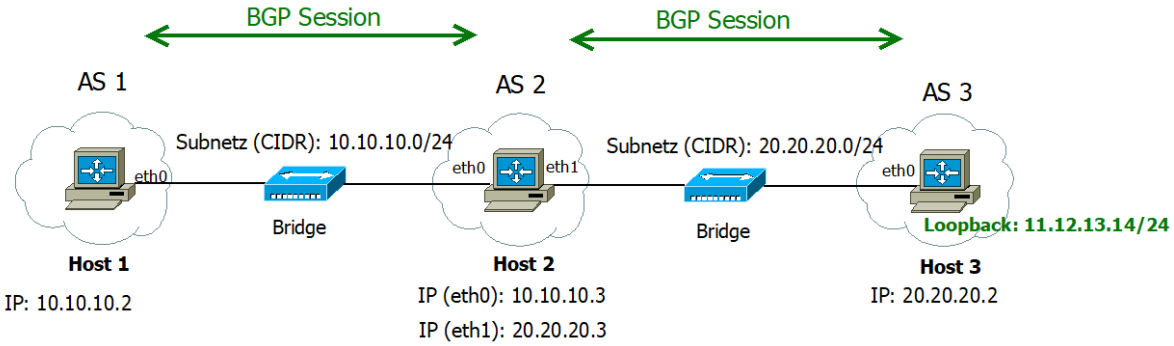
e) Nein, denn sonst würde AS3 als Transit-AS missbraucht werden.

f) Beim Eintippen der IP Adresse stellt man fest, dass die FH Rosenheim keine eigene AS Nummer hat, sondern über das Deutsche Forschungsnetz (DFN) mit der AS Nummer AS1275 ans Internet angebunden ist.

g) Aktuell sind am DE-CIX ca. 919aktive ASe. Das heißt aber nicht, dass jedes Paar von ASen auch tatsächlich Daten austauschen. Durch BGP Routing Policies kann jedes AS explizit entscheiden, mit wem es Daten austauschen möchte. Ein IXP stellt nur eine Art Treffpunkt dar. Wer mit wem redet, kann frei entschieden werden. Manche Asen wie z.B. Verizon veröffentlichen explizit zu welchen Bedingungen sie Verkehr austauschen.

Aufgabe 3: BGP Routing mit Linux

c) IP Adresen, siehe Skizze



d) Das Subnetz 11.12.13.0/24 ist nur in der Routingtabelle von Host3 zu finden. Alle anderen Hosts kennen keine Route zu diesem Netz.

e) Der folgende Screenshot zeigt exemplarisch die Bird-Config von Host 2:

```
# 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.

# Change this into your BIRD router ID. It's a world-wide unique identification
# of your router, usually one of router's IPv4 addresses.
router id 20.20.20.3;

# The Device protocol is not a real routing protocol. It doesn't generate any
# routes and it only serves as a module for getting information about network
# interfaces from the kernel.
protocol device {
}

# The Kernel protocol is not a real routing protocol. Instead of communicating
# with other routers in the network, it performs synchronization of BIRD's
# routing tables with the OS kernel.
protocol kernel {
        metric 64;      # Use explicit kernel route metric to avoid collisions
                        # with non-BIRD routes in the kernel routing table
        learn;
        import none;
        export all;     # Actually insert routes into the kernel routing table
}

# BGP configuration
protocol bgp toAS3 {
        local as 2;
        neighbor 20.20.20.2 as 3;
        import all;
        export all;
}
protocol bgp toAS1 {
        local as 2;
        neighbor 10.10.10.2 as 1;
        import all;
        export all;
}

# restrict network interfaces BIRD works with, receive routes directly from interface
protocol direct {
        interface "lo";
}
```

f) Die Routingtabelle von Host1 und Host2 haben einen Eintrag für 11.12.13.0/24. Host3 hat unter Umständen keinen Eintrag für die Loopbackadresse. Der Ping funktioniert nur von Host2 und Host3. Der Grund: Host3 wüsste nicht, wohin es die Antwort für einen ping von Host1 senden müsste.

g)

h) Circa 1mal pro Minute werden Keepalive Nachrichten gesendet, siehe auch bgp.pcapng.

i) In Paket #40, siehe bgp.pcapng wird der Präfix zurückgezogen („withdraw“).

j) In Paket #50 wird der Präfix erneut angekündigt:

- Pfadattribute: ORIGIN, AS-Pfad, NEXT-HOP.
- NLRI lautet: 11.12.13.0/24
- Host1 kennt wieder die Route zum Zielnetz 11.12.13.0/24

```
root@host1:/# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.10.10.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
11.12.13.0      10.10.10.3      255.255.255.0   UG    64     0        0 eth0
```