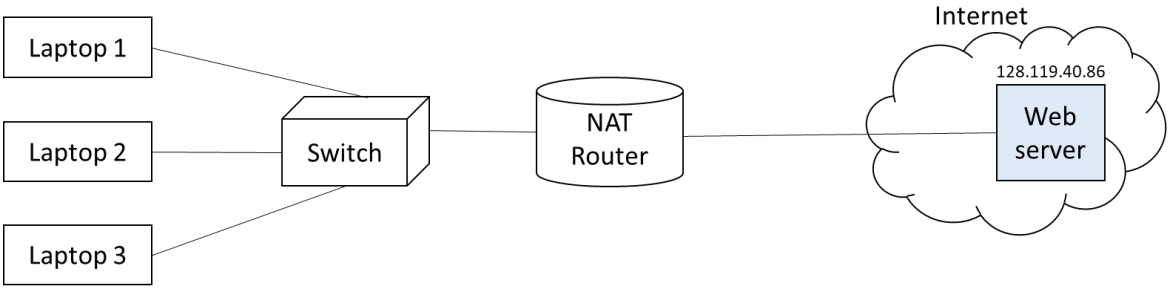
Übung 12: NAT, DNS, HTTP

Aufgabe 1: Network Address Translation (NAT)

3 Laptops in einem Heimnetzwerk sind über einen NAT-fähigen Home Router mit dem Internet verbunden. Die öffentliche IP des Routers ist 24.34.112.235, im Heimnetzwerk dürfen ***nur*** IP-Adressen aus dem Bereich 192.168.0.0/24 gewählt werden.



a) Weisen Sie *allen* Interfaces innerhalb des Heimnetzwerkes manuell eine gültige IP-Adresse zu. Tragen Sie auch die öffentliche IP-Adresse des Routers in die Zeichnung ein.

b) Zu einem bestimmten Zeitpunkt unterhalten alle Laptops im Heimnetzwerk gleichzeitig je 2 HTTP Verbindungen zum Web Server. Wie könnten eine mögliche NAT-Tabelle des Home Routers für diesen Zeitpunkt aussehen?

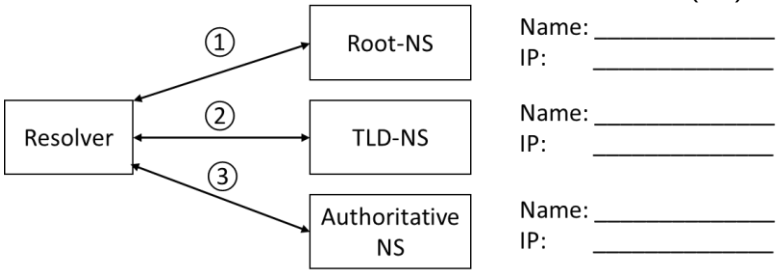
LAN Seite/Heim-Netzwerk		***WAN Seite / Internet***	
IP Adresse	Port	IP Adresse	Port

Aufgabe 2: DNS

In dieser Aufgabe wird **`dig`**[1] verwendet, um DNS-Anfragen zu stellen. Das Tool ist in der Linux-VM bereits vorhanden.

a) Finden Sie die IP Adresse Ihres ***lokalen*** DNS Resolvers heraus! *Tipp*: Datei `/etc/resolv.conf`

b) Spielen Sie „DNS Resolver“ und finden Sie durch ***manuelle, iterative*** DNS-Anfragen die IP Adresse des Webservers der TH Rosenheim (**`www.th-rosenheim.de`**) heraus. Sie dürfen **nicht** den lokalen Nameserver (NS) direkt fragen.



[1] Alternative für Windows und Linux: `nslookup`

1) Sie beginnen beim Root-NS: Ermitteln Sie den NS der Top-Level-Domin(TLD).
2) Fragen Sie dann den TLD-NS.
3) Fragen Sie dann den NS der TH Rosenheim nach der IP des Webservers.

Tragen Sie jeweils den Namen und die IP Adresse der Nameservers in die Grafik ein. Was ist die finale Antwort? Hinweise:

- Verwenden Sie jeweils **`dig @<NS-IP> <NAME>`** um den Nameserver mit der IP `<NS-IP>` nach dem Namen `<NAME>` zu befragen.
- IP-Adressen der Root-NS: https://de.wikipedia.org/wiki/Root-Nameserver

c) Welche IP Adresse hat der Mailserver der Domain `th-rosenheim.de`?
Tipp: `dig MX th-rosenheim.de` und dann eine weitere Anfrage stellen.

d) Verwenden Sie `dig ANY <NAME>`, um über Ihren Standard-DNS-*Resolver* sowohl die IPv6 Adresse der Webseite www.muenchen.de als auch der Webseite www.berlin.de herauszufinden. Sind die beiden Webserver bereits IPv6-fähig?

e) Welche IP Adresse hat `www.microsoft.de`. Warum werden hier mehrere IP Adressen zurückgegeben? Ändert sich die Reihenfolge bei einer erneuten Anfrage? Falls ja, warum?

f) Zeichnen Sie unter Wireshark[2] auf, während Sie mit `dig` einen exotischen Namen auflösen. „Exotisch" bedeutet, dass die dazugehörige IP gerade nicht im Cache des Resolvers sein sollte. Suchen Sie dann im Wireshark-Trace die dazugehörige DNS Anfrage und Antwort.

- Was ist der Standard-Port von DNS?
- Was wird außer der IP Adresse ggfs. noch zurückgeliefert?

Aufgabe 3: HTTP

Hinweis: Die Teilaufgaben f) bis g) erfordern Wissen, das erst in der Vorlesung am Dienstag besprochen wird. Ggfs. vorerst weglassen.

Der bereitgestellte Wireshark-Trace `https.pcapng` enthält das folgende Szenario:

- Abruf der Webseite www.bayern.de
- Zwischenzeitliches Laden einer anderen Seite
- Erneuter Abruf der Webseite www.bayern.de (Paket #4920)
- Browser Refresh (Strg+R): Erneutes Laden von www.bayern.de (Paket #5221)

a) Gleich zu Beginn des Traces wird eine TCP Verbindung aufgebaut. Welche IP Adresse hat der TCP Server und TCP Client? Welche Ports werden verwendet?

b) Filtern Sie nach HTTP Paketen? Warum sehen Sie nichts?

c) Wireshark: *Bearbeiten-Einstellungen-Protocols-TLS* → Unter „*Pre-Master Secret log filename*" die mitgelieferte Datei „`keys.log`" einstellen. Sehen Sie nun HTTP Pakete?

d) HTTP Request: Finden Sie die erste Abfrage der Webseite www.bayern.de
- Paketnummer in Wireshark?
- Welche Protokolle werden auf welchen Schichten eingesetzt?
- Welche HTTP Version verwendet der Web Browser?
- Was ist die bevorzugte Sprache für die Webseite, die der Server ausliefern soll?

e) HTTP Response: Finden Sie die Antwort des Webservers auf die Anfrage von d).
- Paketnummer in Wireshark?
- Welchen Status Code liefert der Web Server zurück?
- Wann wurde die ausgelieferte HTML Datei das letzte Mal auf dem Server verändert?
- Wieviel Bytes benötigt der HTTP Header?
- Wie viele TLS Segmente wurden benötigt?

[2] Wireshark ggfs. als Admin starten.

f) Pipelining, Persistent Connections: Nach dem initialen `HTTP GET` und `RESPONSE`, siehe d) und e) werden zahlreiche weitere Ressourcen angefordert, z.B. `css`-Dateien, `js`-Dateien. Betrachten Sie nur `HTTP` Pakete (Filter!), es genügen wenn Sie alle HTTP Pakete bis #127 anschauen:
   - Handelt es sich um Persistent HTTP?
   - Bei Firefox: Finden sie in `about:config` heraus, wie viele parallele persistente TCP Verbindungen Ihr Browser maximal zulässt zu einem Server zulässt? Filter: „*connections*“

g) Cookies: Der erste Aufruf, siehe d), erfolgte bei leerem Cookie-Cache des Browsers. Anschließend wird in Paket #4920 die Webseite erneut aufgerufen.
   - Setzt das 200 OK im Paket #31 einen Cookie?
   - Wird beim 2. Aufruf (Paket #4920) das Cookie durch den Webbrowser an den Webserver gesendet.
   - Wo können Sie sich in Ihrem Browser die gespeicherten Cookies ansehen?

h) Bedingtes GET: Prinzipiell müssten alle Ressourcen bei einem Browser-Refresh („`Strg+R`“) erneut geladen werden. Geschieht das? Erklären Sie exemplarisch anhand des HTTP Requests #5307 und der Antwort in #5339.