FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Practical Computer Networks and Applications

Exercise 2 – IP Version 6 Networks

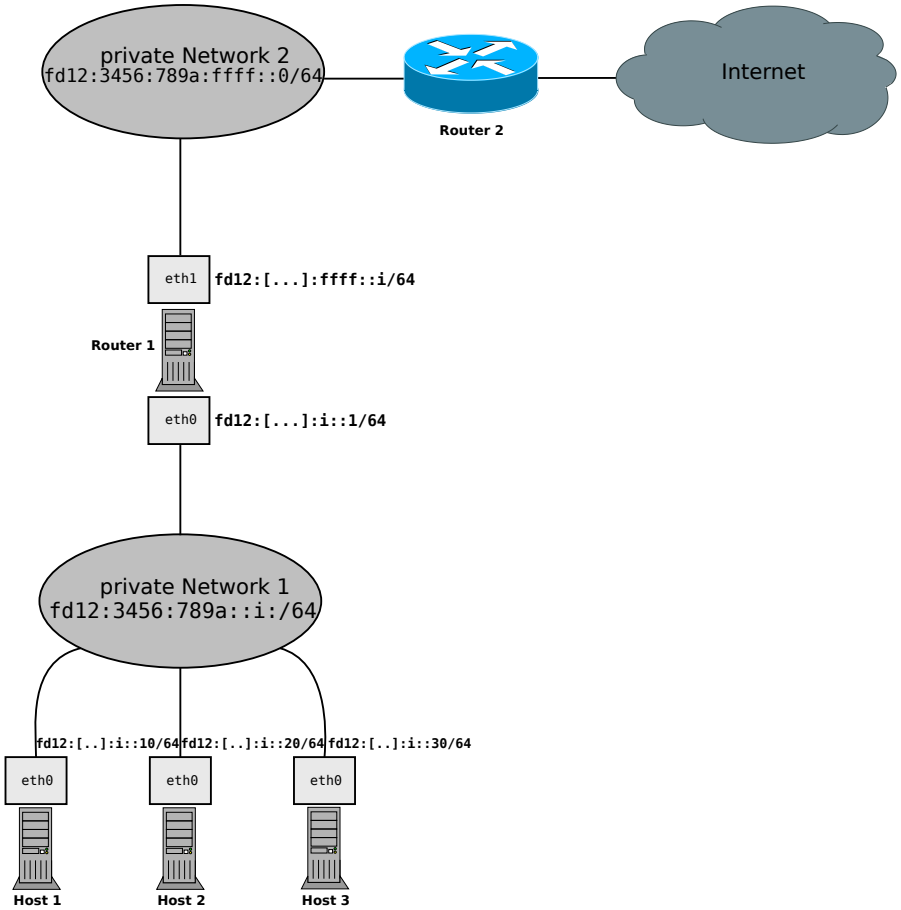
Prof. Dr. Baun, Prof. Dr. Ebinger, Prof. Dr. Hahm, Prof. Dr. Kappes, Dipl. Inf. (FH) Maurizio Petrozziello

{baun,peter.ebinger,oliver.hahm,kappes,petrozziello}@fb2.fra-uas.de

Frankfurt University of Applied Sciences
Faculty of Computer Science and Engineering
Nibelungenplatz 1
60318 Frankfurt am Main

Contents

Network Topology – Exercise 2 – ULA



Network Topology of lab exercise 2 – Task 2 ULA (Unique Local Address)

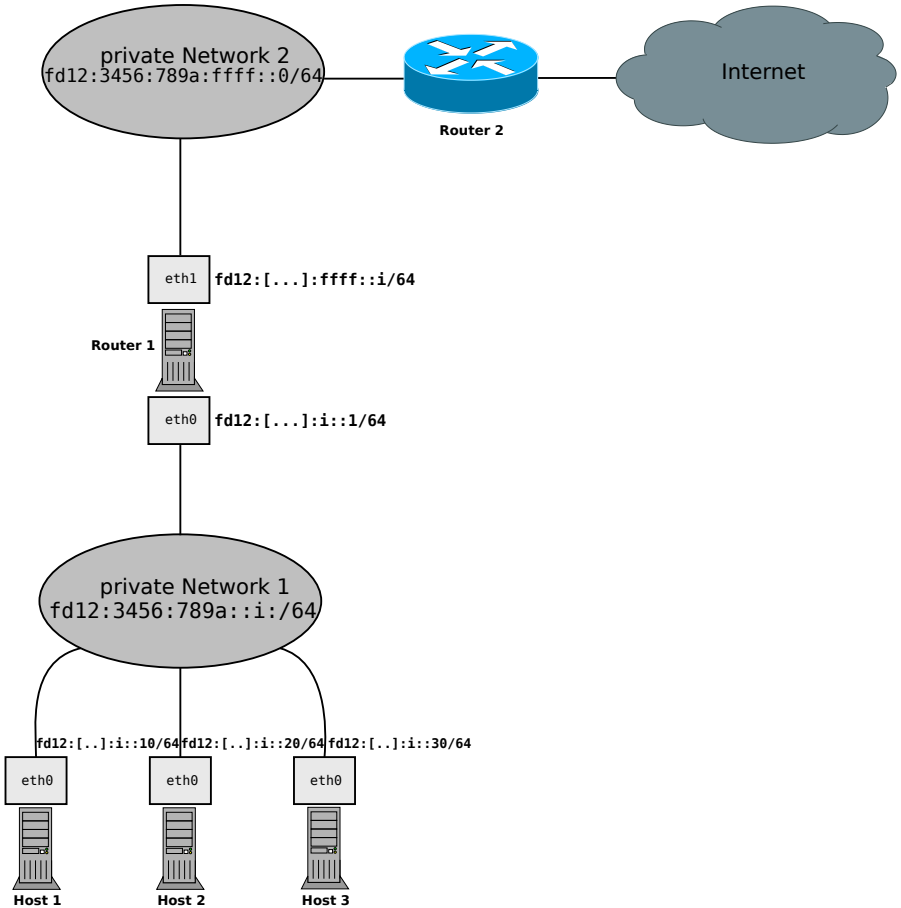
Private Network 1:

- `fd12:3456:789a:i::0/64`
- Private Network for host machines and Routers
- The number `i` in the IP address is a placeholder for your **group number**!

Private Network 2:

- `fd12:3456:789a:ffff::0/64`
- Private network connecting all networks

Network Topology – Private Network 2



Network Topology of lab exercise 2 – Task 2 ULA (Unique Local Address)

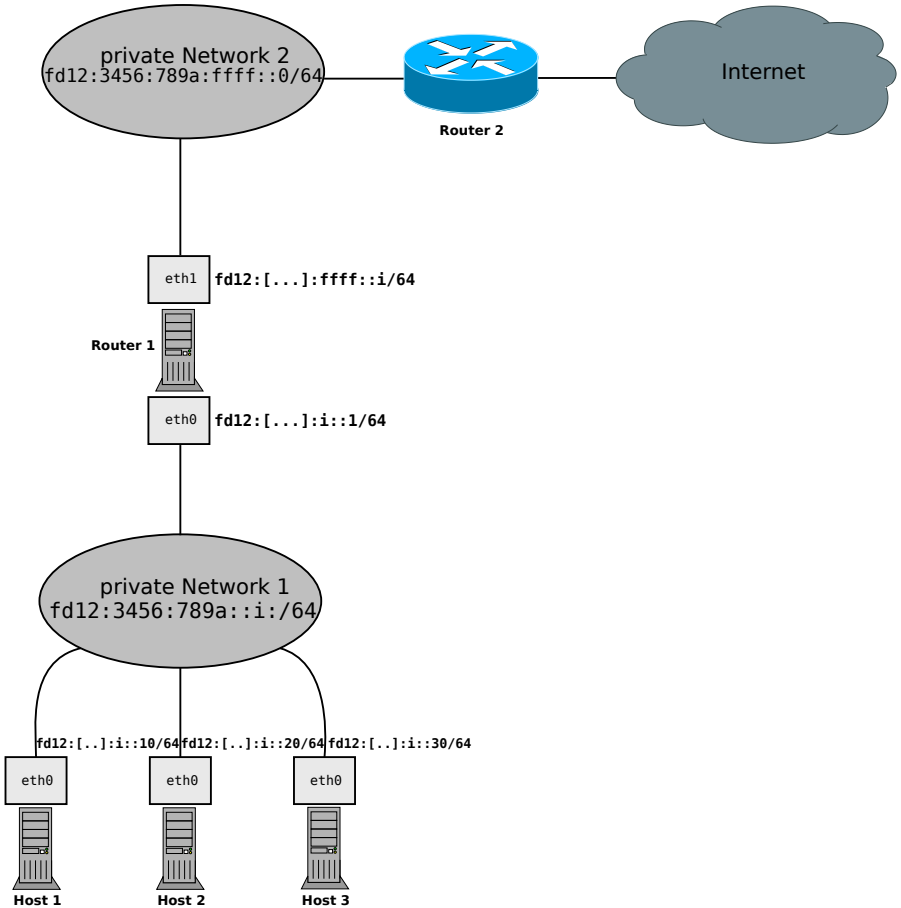
Private Network 2:

- `fd12:3456:789a:ffff::0/64`

Router 2:

- `fd12:3456:789a:ffff::ffff`
- **Router 2** is the gateway for all Routers in private network 1!
- The route to **Router 2** needs to be configured on **Router 1**!
- **Router 2** runs a web server on **port 80**!

Network Topology – Private Network 1



Network Topology of lab exercise 2 – Task 2 ULA (Unique Local Address)

Private Network 1:

- `fd12:3456:789a:i::0/64`

Router 1:

- Has two interfaces
- `eth0`: `fd12:3456:789a:i::1`
- `eth1`: `fd12:3456:789a:ffff::i`

Host Network:

- **Router 1**: `fd12:3456:789a:i::1`
- **Host 1**: `fd12:3456:789a:i::10`
- **Host 2**: `fd12:3456:789a:i::20`
- **Host 3**: `fd12:3456:789a:i::30`

Network Topology – Lab Exercise 2 – Objectives

In the lab exercise you need to accomplish...

- a successful static configuration of the machines!
- successful autoconfiguration of the hosts!
- working static routing on the machines!
- reachability of all machines (all hosts including **Router 1** and **2**)!

Contents

Exercise 2

IP Version 6 Addresses

IPv6 Addresses

Tabelle: IPv6 address ranges

IPv6 Address	Purpose
`2001:db8::/32`	Documentation prefix used for examples
`::1`	Localhost
`fc00::/7`	Unique Local Addresses (ULA) also known as ”Private” IPv6 addresses. (Currently not used! See source: RFC 4193 Section 3.2)
`fd00::/8`	Unique Local Addresses (ULA) **L-bit** set to **1** for local IPv6 address prefix
`fe80::/10`	Link Local addresses, only valid inside a single broadcast domain
`2001::/16`	Global Unique Addresses (GUA) Routable IPv6 addresses
`ff00::0/8`	Multicast addresses

IPv6 Addresses

Multicast-Scope – `ff00::0/8` multicast groups for specific services in a network. Starting with `ff` and followed by 4 flag bits and 4 bits for indicating specific services.

- Scope `ff01` is for local interface (not leaving interface)
- Scope `ff02` is for link local address space

In addition to the scope, a Multicast group is specified:

`ff0X::1` : All IPv6 stations
`ff0X::2` : All Routers
`ff0X::f` : UPnP
`ff0X::101` : All Timeservers (NTP)
`ff0X::1:2` : DHCPv6 Server

Not all Multicast groups make sense for all Multicast scopes

IPv6 Addresses

There are four way of configuring IPv6 addresses:

- Static adressing with ULA (RFC 4193)
- SLAAC (RFC 4862)
- Stable Private (RFC 7217)
- Privacy Extension (RFC 4941)

IPv6 Addresses in Linux

The `ip`[1] command:

- `ip addr ...` – configuration of IPv6 addresses
- `ip route ...` – configuration of IPv6 routes

`-6` option in `ip`

The option `-6` specifies the use of IPv6 addresses. It is important to use this option because without the parameter `ip` defaults to IPv4!

[1] The manpage of `ip` gives you the full list of functions and options!

Enabling IPv6 Addresses in Linux

Enable IPv6:

```
# sysctl -w net.ipv6.conf.all.disable_ipv6=0
# sysctl -w net.ipv6.conf.lo.disable_ipv6=0
# sysctl -w net.ipv6.conf.default.disable_ipv6=0
```

The file `sysctl.conf` and command `sysctl`

Kernel parameters (as enabling IPv6 e.g.) can be set with the command `sysctl`[a]! To make the changes permanent you need to edit the file `sysctl.conf`! In the lab the use of `sysctl` is sufficient, since changes will be overwritten after reboot!

[a] More information: `https://linux.die.net/man/8/sysctl`

Static IPv6 Unique Local Addresses – ULA

Tabelle: RFC 4193 Addressing Scheme

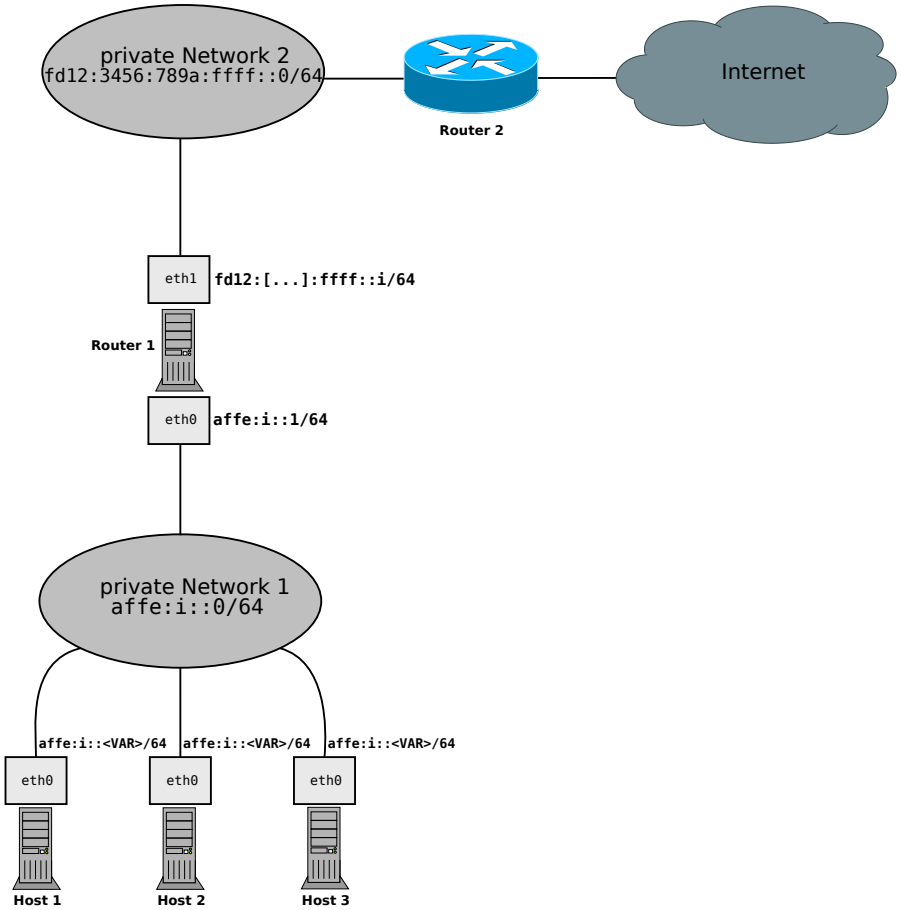
Prefix/L	Global ID	Subnet ID	Interface ID
fd00::/8	**40 bits**	**16 bits**	**64 bits**
fd00::/8	12:3456:789a	:0001	0000:0000:0000:0001

Resulting IPv6 address: fd12:3456:789a:0001:0000:0000:0000:0001
Short IPv6 address: fd12:3456:789a:1::1

Tabelle: RFC 4193 Lab Exercise 2

Machine	Prefix/L	Global ID	Subnet ID	Interface ID
Router 1	fd00::/8	xx:xxxx:xxxx	**i**	0000:0000:0000:0001
Host 1	fd00::/8	xx:xxxx:xxxx	**i**	0000:0000:0000:0010
Host 2	fd00::/8	xx:xxxx:xxxx	**i**	0000:0000:0000:0020
Host 3	fd00::/8	xx:xxxx:xxxx	**i**	0000:0000:0000:0030

Network Topology – Exercise 2 – Autoconfiguration



Network Topology of lab exercise 2 – Task 3 Autoconfiguration

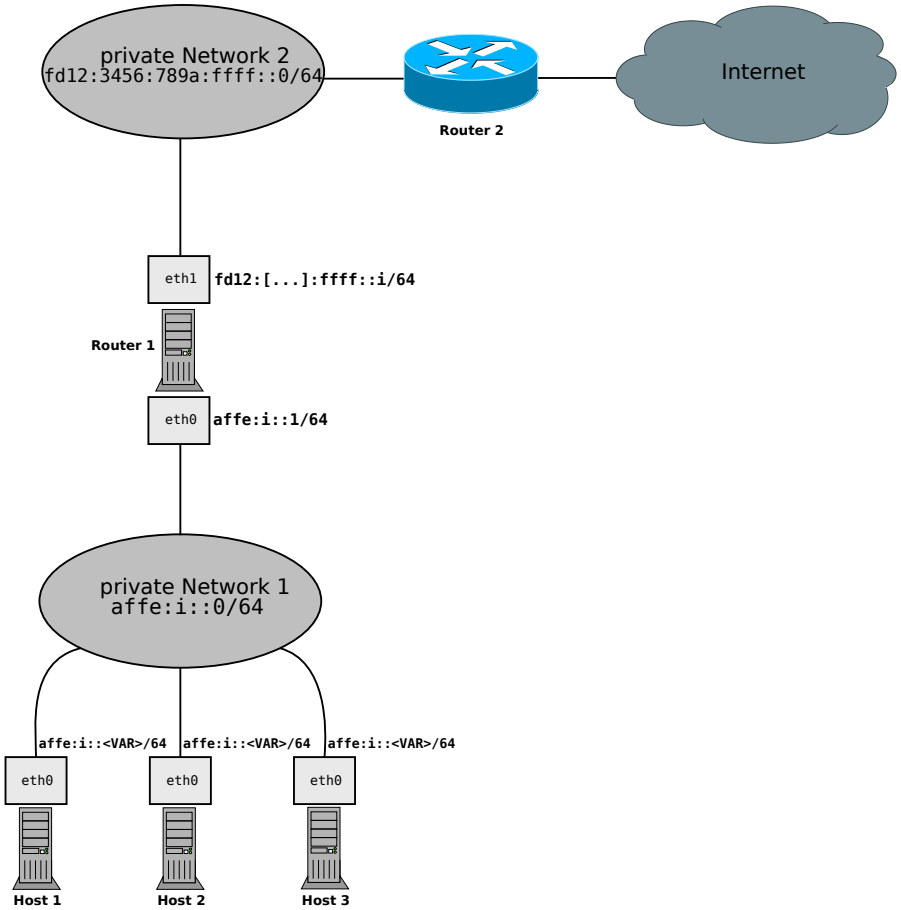
Private Network 1:

- `affe:i::0/64`
- Private Network for host machines and Routers‘

Private Network 2:

- `fd12:3456:789a:ffff::0/64`
- Private Network spanning all networks

Network Topology – Private Network 2



Network Topology of lab exercise 2 – Task 3 Autoconfiguration

Private Network 2:

- `fd12:3456:789a:ffff::0/64`

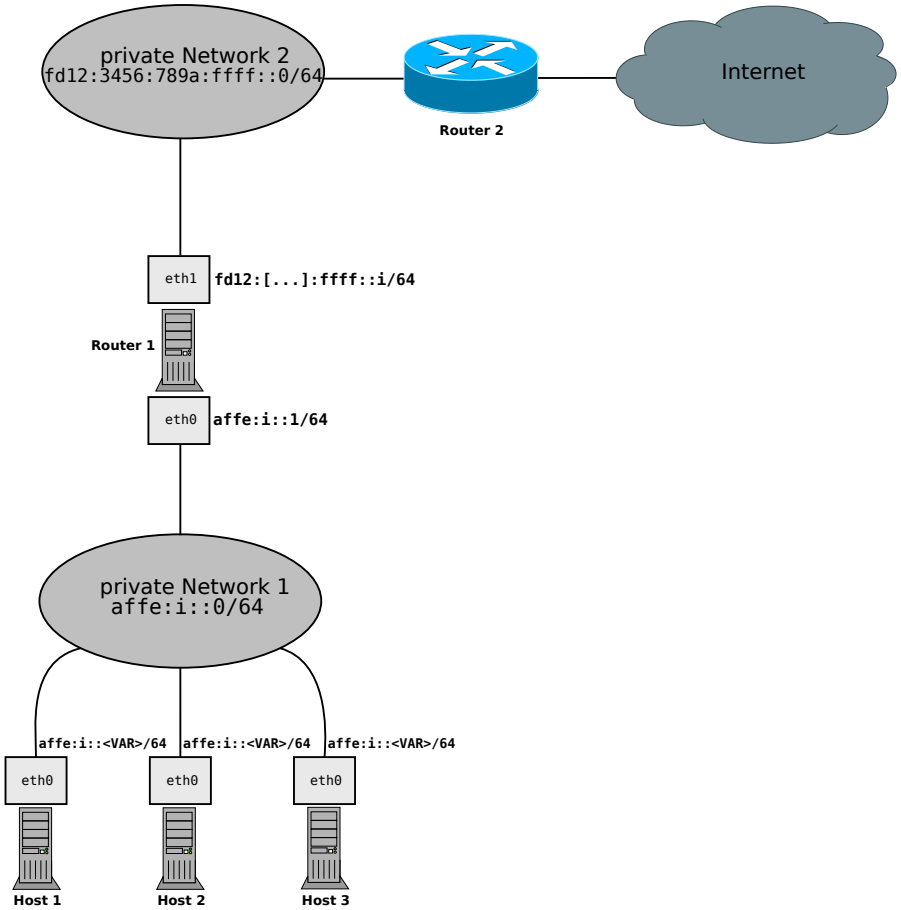
Router 2:

- `fd12:3456:789a:ffff::ffff`

Router 2 is the gateway for all Routers in private network 1!

The route to **Router 2** needs to be configured on **Router 1**!

Network Topology – Private Network 1



Network Topology of lab exercise 2 – Task 3 Autoconfiguration

Private Network 1:

- `affe:i::0/64`

Router 1:

- Has two interfaces
- `eth0: affe:i::0/64`
- `eth1: fd12:3456:789a:ffff::i`

Host Network:

- **Router 1** – `affe:i::1`
- **Host 1** – `affe:i::<VAR>`
- **Host 2** – `affe:i::<VAR>`
- **Host 3** – `affe:i::<VAR>`
- `<VAR>` is the placeholder for a dynamically generated address!

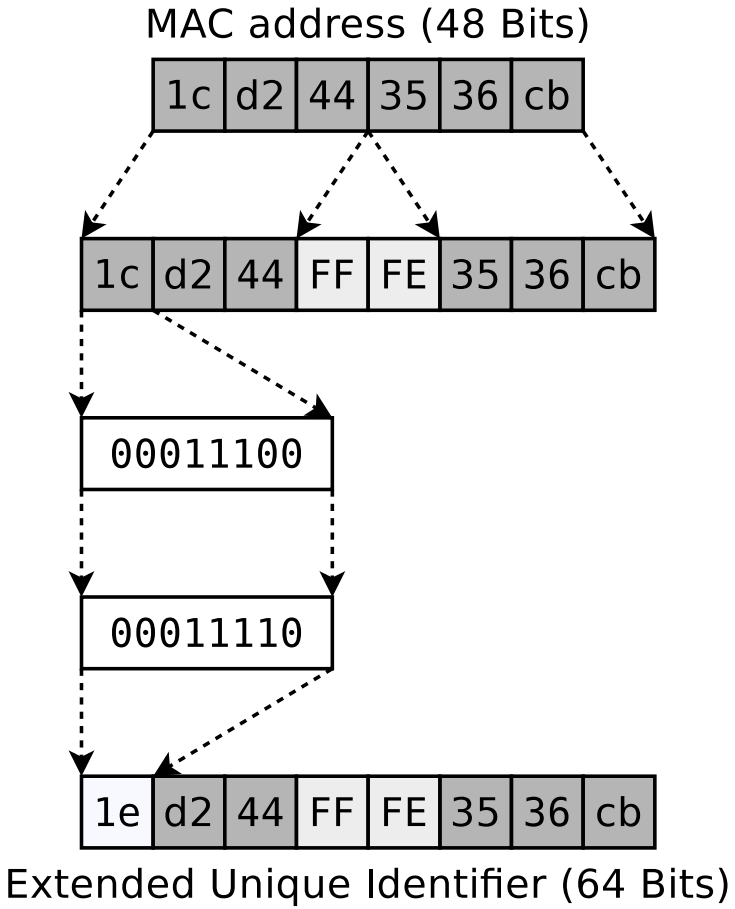
Stateless Address Autoconfiguration – SLAAC (RFC 4862)

- The RFC 4862 defines the automatic stateless address generation
- The Host uses its MAC address for the generation of the 64-bit Host-ID (**EUI-64**)
- The Network Prefix is defined by the scope and or the Router (e.g. `fe80::/64` for link-local)
- **Benefit** $\rightarrow$ Stateless generation without an external Router

Router Advertisement Daemon (`radvd`)

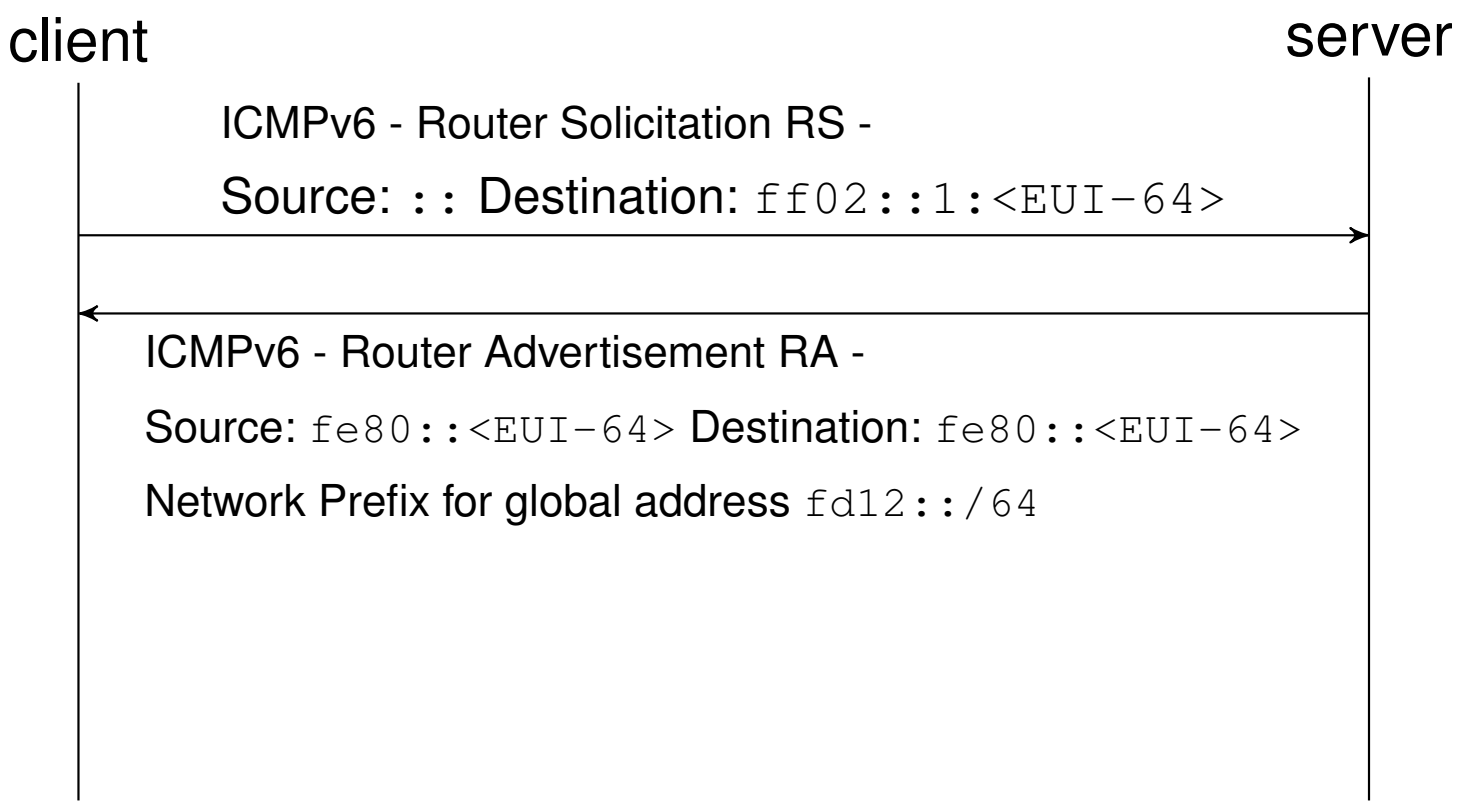
For the automatic assignment of Network prefixes the Router needs a `radvd` for the management of network prefixes in the network. Without `radvd` the link local prefix `fe80::/64` is used!

Stateless Address Autoconfiguration – SLAAC (RFC 4862)



EUI-64 calculation

Stateless Address Autoconfiguration – SLAAC (RFC 4862)



Message Sequence Diagramm for Router Solicitation

Stable Privacy – RFC 7217

- The RFC 7217 defines the address generation without the use of a MAC address
- A random secret key is generated and used for the generation of the Interface-ID
- Once generated, the Interface-ID is assigned and does not change anymore (until reboot!)
- **Benefit**: Increased security because no MAC address is used for generation!

Secret Key and Kernel parameter

The stable secret value is stored in the directory `/proc/sys/net/ipv6/conf/eth0/stable_secret` and is generated by setting the Kernel parameter `addr_gen_mode=3`!

Stable Privacy – RFC 7217

Example of a generated stable private address:

```
MAC: 86:3a:ea:8a:a7:d9
stable-privacy -> inet6 fe80::6f6d:80e:ab6c:65a0/64
link local -> inet6 fe80::843a:eaff:fe8a:a7d9/64
```

Example of `stable secret` parameter:

```
$ cat /proc/sys/net/ipv6/conf/eth0/stable_secret
c8c8:036d:9312:71e2:eadc:7c9f:0535:649a
```

Stable Privacy – RFC 7217

In contrast to SLAAC RFC 7217 brings the following benefits:

- \+ Host's MAC address is not exposed!
- \+ The address is stable for the Host

Privacy Extension – RFC 4941

- RFC 4941 defines the address generation with a random number
- It is using the address in a temporary manner
- A new Interface-ID gets generated periodically
- Old Interface-IDs can still be used for established connections
- **Benefit**: Increased security because no MAC address is used for generation!
- **Drawback**: Address is not stable!

Random generation of Interface-ID

RFC 4941 defines a scheme for the generation of addresses where values for the lifetime are defined and the valid lifetime is calculated with the formula:

`CREATION_TIME + TEMP_PREFERRED_LIFETIME - DESYNC_FACTOR`

Where `CREATION_TIME` is the time at which the address was created, `TEMP_PREFERRED_LIFETIME` (the maximum time of validity) and `DESYNC_FACTOR` (a random number in the range of 0 to 600 seconds)!

Source: `https://datatracker.ietf.org/doc/html/rfc4941#page-13`

Privacy Extension – RFC 4941

Example of a random generated address:

```
MAC: 86:3a:ea:8a:a7:d9
privacy-extension -> inet6 fd12::8992:3c03:d6e2:ed72/64
link local -> inet6 fe80::843a:eaff:fe8a:a7d9/64
```

Random generation of Interface-ID

The address shown above is generated randomly and temporary and cannot be traced back to any host characteristics!

Privacy Extension – RFC 4941

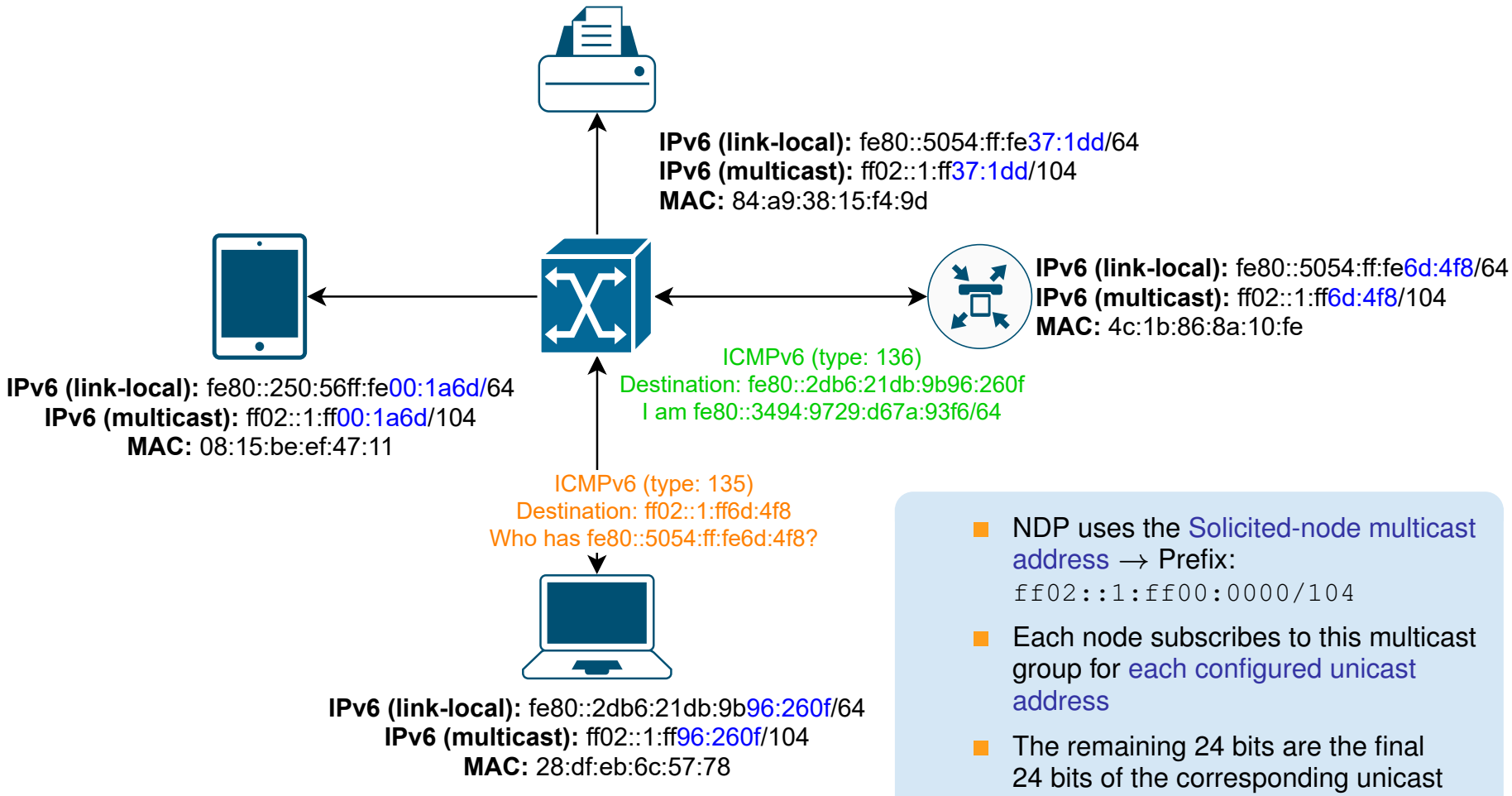
Compared to SLAAC, the RFC 4941 method has these benefits:

+ Host's MAC address is not exposed!
+ The address is generated dynamically over time!
- **Benefit**: Increased security because no MAC address is used for generation!

Compared with Stable Privacy, the RFC 4941 method has these benefits:

+ Host address is changed over time, therefore increased security!
- **Benefit**: Increased security because address expires!
- **Drawback**: Address is not stable!

Neighbor Discovery Protocol – NDP



- NDP uses the Solicited-node multicast address → Prefix: `ff02::1:ff00:0000/104`
- Each node subscribes to this multicast group for each configured unicast address
- The remaining 24 bits are the final 24 bits of the corresponding unicast address
- Only nodes registered to this address will receive the ICMP message

Configuration of the machines

Please follow these rules:

- **Make your configurations statically! Use the tool `ip` exclusively!**
- **Save your static configuration on file! Use an USB-Drive for the extraction!**
- **Test your setup! Document it accurately! Demonstrate it in the lab exercise!**
- **Create slides of your configurations! Use the command-line snippets, screenshots and Wireshark captures for your documentation!**

Non persistent configuration on machines

Please be aware, that the configurations on the machines are static and will be deleted after a reboot! Make sure to save your progress on an external drive!