1. Lab information

Network Topology

The network topology is shown in Figure 1. PCs belongs to two different Ethernet segments.

The address space is spitted as follows:

- **Segment A**: 2 PCs (virtual machines) + Wifi laptops
  - 2001:4b58:27:aaaa::/64 / 194.141.27.224/28
- **Segment B**: 6 PCs (virtual machines)
  - 2001:4b58:27:aaaa::/64
  - 194.141.27.240/28

![Figure 1: Lab topology](image)
Exercise A: Enable IPv6 to WinXP

Objectives

- Activate the IPv6 protocol stack on WinXP PC’s
- Understand basic IPv6 concepts
- Manually add/remove IPv6 addresses on Win XP
- Disable 6to4 and isatap virtual interfaces

Exercises steps

1. There are two alternative methods for activation IPv6 in WinXP (SP2)
   - Use the WinXP GUI to install the new protocol
     - From a CLI run “ipv6 install”
   - Use the WinXP GUI to install the new protocol

2. Identify the available interface at your PC. Identify which of these interfaces are related to IPv6 transition mechanisms? From a CLI run the following commands
   - ipconfig /all
   - netsh interface ipv6 show interface
   - ipv6 -v if

3. Identify all the IPv6 addresses (link local, public addresses, etc)
   - Link local (Tip: Search for fe80::...)
   - Identify the auto-configuration IPv6 address (Tip: Search for ...ff:fe...)
   - Identify the IPv6 address due to privacy extension
   - Identify the validity of addresses (Tip: Use the command netsh interface ipv6 show address <interface>)

4. Ping / traceroute IPv6 hosts
   - Ping the IPv6 localhost address (::1)
   - Ping other addresses
   - Ping IPv6 web sites (www.grnet.gr, www.6diss.org, etc)

5. Find IPv6 neighbours in your LAN. What could be the problem in terms of security?
   - (Tip: Use the command netsh interface ipv6 show neighbors)

6. Identify the local router address.
• What is the appropriate command? “traceroute”? “... show neighbours”?

7. Use “ethereal” tool to capture IPv6 traffic, e.g. advertisements (RAs), or own traffic. Which IPv6 address is used when communicating?
   • Tip: See at the end of the document for ethereal filters.

8. Disable privacy extensions (RFC3041). What could be the problem in terms of security if you enable / disable privacy extension?
   • (Tip: Use the command `netsh interface ipv6 set privacy ...

9. Disable 6to4 and isatap virtual interfaces.
   • (Tip: Use the command `netsh interface ipv6 6to4 state ...

...
Exercise B: Enable IPv6 to Linux

Objectives
In this laboratory exercise you will complete the following tasks:

- Check for IPv6 support in the running kernel
- Understand basic IPv6 concepts
- Manually add/remove IPv6 addresses on Linux
- Use some basic IPv6 related tools

Task 1: Verify IPv6 support in your Linux

Complete the following exercise’s steps:

Step 1: Check for IPv6 support in the running kernel.
Modern Linux distributions already contain IPv6-ready kernels, the IPv6 capability is generally compiled as a module. To check whether your current running kernel supports, or not, IPv6 the following file must exist: `/proc/net/if_inet6`

It’s possible that the IPv6 module is not loaded automatically on startup. So, verify that the module is running by listing the current loaded modules:

```
lsmod | grep ipv6
```

Task 2: Display and identify existing IPv6 addresses

Complete the following exercise’s steps:

Step 1: Check, if and which IPv6 addresses are already configured

Run the following commands:

- `ip` command
  ```
ip -6 addr show dev <interface>
  ```
- `ifconfig` command
  ```
  ifconfig <interface>
  ```
- `netstat` command
  ```
  netstat -inet6 -g
  ```

Step 2: Using the previous commands, identify the different types of IPv6 addresses

- Link local (*Tip: Search for fe80::...*)
- Auto-configuration IPv6 address (*Tip: Search for ff:ee...*)
- Multicast address
- Validity of addresses
- Can you identify any IPv6 address due to privacy extension?
Task 3: Using some IPv6 related tools

Complete the following exercise’s steps:

**Step 1:** Ping IPv6 addresses
- Ping the IPv6 localhost address (::1)
- Ping your host’s link-local and global addresses
- Ping your host’s multicast addresses (*Tip:* Use the option -I in ping6 command)

**Step 2:** Without looking into the router, identify the router’s link-local address for your VLAN (*Tip:* Use the command `ip -6 neigh ...*)
- Ping router’s addresses (link-local and global addresses). Did you successfully ping router’s link-local address? (*Tip:* Use the option -I in ping6 command)

**Step 3:** Using tcpdump (`tcpdump -t -n -vv -s 512 ip6 -i <Interface>`), capture router advertisement and router solicitation messages. If you want, look at Appendix A for tcpdump basic information or use your linux manual pages. If you have ethereal on your Linux you can use it instead of tcpdump.
- Which IPv6 addresses (source - destination) are used in this messages?
- Open a second console. In this new console restart your network (`/etc/init.d/network restart`). Quickly change to the other console and execute the tcpdump command. You should have captured one router solicitation message. Look at the first router advertisement message after the RS message. What’s the source/destination address? Are they equal to the previous RA message? Why?

**Step 4:** Display your current IPv6 routing table (*Tip:* Use command `route -A inet6 -n`). Identify the next hop for your default gateway.

Task 4: Add/Remove IPv6 addresses

**Step 1:** Manually add an IPv6 address
On your network interface, add the following address
 e.g: 2001:4b58:27:aaaa::10
(Note: DAD should occur if more than one machines tries to use )

You can accomplish this task using different methods:

- Using ip command (temporary address).
  `ip -6 addr add <ipv6address>/<prefixlength> dev <interface>`

- Using ifconfig command (temporary address).
  `ifconfig <interface> inet6 add <ipv6address>/<prefixlength>`
• **Adding address to system configuration:** This depends on the Linux distribution used – It may be in one of the following files:
  - Fedora: `/etc/sysconfig/network-scripts/ifcfg-eth(X)` and adding an entry similar to:
    ```
    IPV6ADDR=address/prefixlentgh
    ```
  - Debian/Ubuntu: `/etc/network/interfaces` and adding:
    ```
    iface eth0 inet6 static
    address 2001:0db8:0005:0006::78
    netmask 64
    ```

• Then you’ll need to restart your network `/etc/init.d/network restart`

**Step 2:** Manually remove an IPv6 address
Remove the address created on the previous step.
You can accomplish this task using different methods:

- Using `ip` command.
  ```
  ip -6 addr del <ipv6address>/<prefixlength> dev <interface>
  ```

- Using `ifconfig` command.
  ```
  ifconfig <interface> inet6 del <ipv6address>/<prefixlength>
  ```

- **Removing address from system configuration** – Remove/comment out entries added in Step 1, then restart your network `/etc/init.d/network restart`

**Step 3:** Enable privacy extensions (RFC3041) or temporary addresses.
To complete this step you have to modify the kernel’s running setting `net.ipv6.conf.default.use_tempaddr`. (Please, consult the appendix B for more details). You can achieve this by doing one of these commands:

- `sysctl net.ipv6.conf.default.use_tempaddr=V`
  Or
- `echo "V">/proc/sys/net/ipv6/conf/default/use_tempaddr`

Where $V$ is an integer (see Appendix D)

Then fill the table below:

<table>
<thead>
<tr>
<th>Value for $V$</th>
<th>Privacy extensions enabled? (Yes/No)</th>
<th>Preferred address (temporary/global)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 1: exercise’s table

Tip: Probably the easiest way to fill the table is doing the following steps for all the three values (2, 1, 0):

- `sysctl net.ipv6.conf.default.use_tempaddr=V`
- Restart your network (/etc/init.d/network restart)
- With `ifconfig` command check if you have a temporary address
- Ping your routers global address: `ping6 -I <interface> 2001:4b58:27:XY:router_number)` and check what’s the source address
Appendix A: Compact “Ethereal/Wireshark” documentation

Ethereal is used by network professionals around the world for troubleshooting, protocol analysis, software and protocol development. Its open source license allows talented experts in the networking community to add enhancements. It runs on all popular computing platforms, including Linux, and Windows. See further information at http://www.wireshark.com/.

In order to capture packets, use the menu (Capture -> Interfaces…). Then choose the interface you want to use and click on the correspondent Prepare button. In the Capture Options window check both “Update list of packets in real time” and “Automatic scrolling in live capture”. Then uncheck both “Enable transport name resolution” and “Enable MAC name resolution” (see figure 3).

![Ethereal capture options](image)

Figure 2: Ethereal capture options

If you want to capture only a specific set of packets, use Capture Filter option (in the Capture Options window), as shown in Figure 4. Use the capture filter ipv6 (some Ethereal versions use ip6) to capture only IPv6 packets or icmpv6 to capture only ICMPv6 packets.
After having captured some traffic, you can also filter the results using the Filter option, as shown in the Figure 5.

(Tip: Use the filter `ipv6` (some Ethereal/Wireshark’s versions uses `ip6`) to show only IPv6 packets, `icmpv6.code==0` to show ICMP packets of specific code or `http` to show HTTP traffic.)
Appendix B: Using Netsh

Netsh is a command-line scripting utility, for the Windows Operating System, which allows you to display or modify a computer’s network configuration currently running.

Netsh contexts

To run a netsh command, you must start netsh from the CLI prompt and change to the context that contains the command you want to use. The contexts that are available to you depend on which networking components you have installed. For example, if you type dhcp at the Netsh command prompt, you change to the DHCP context, but if you do not have DHCP installed the following message appears:

The following command was not found: dhcp.

Running Netsh commands from the Netsh.exe command prompt

Netsh uses the following standard commands in all contexts that you can run from a Netsh.exe command prompt (that is, netsh>). There might be functional differences between Netsh context commands on Windows 2003 and Windows XP.

1. To view the command syntax, click a command.
2. .. Moves to the context that is one level up.
3. {/?|?|help|h} Displays help at the command prompt.
4. Abort Discards any changes made in offline mode. Abort has no effect in online mode.
5. quit Exits Netsh.exe

Example

The following sample script changes a context from the root context to the interface ipv6 context and adds an IPv6 address:

C:\> netsh
netsh>
netsh> interface ipv6
netsh interface ipv6>
netsh interface ipv6> add address interface=7 address=2001:ABBA::1

Appendix C

Using “IPv6 tcpdump”

On Linux, tcpdump is the major tool for packet capturing. Below you can find some examples. IPv6 support in tcpdump is available since version 3.6. tcpdump uses expressions for filtering packets to minimize the “noise”:
- `icmp6`: filters native ICMPv6 traffic
- `ip6`: filters native IPv6 traffic (including ICMPv6)
- `proto ipv6`: filters tunneled IPv6 in–IPv4 traffic
- `not port ssh`: to suppress displaying SSH packets for running tcpdump in a remote SSH session

Some more command line options are very useful to catch and print more information in a packet, mostly interesting for digging into ICMPv6 packets:

- `-s 512`: increase the snap length during capturing of a packet to 512 bytes
- `-n`: don’t convert host addresses to names.
- `-i`: Listen on `<interface>`
- `-vv`: Even more verbose output

**Example: IPv6 ping to 2001:DB8:CAFE:28::2 native over a local link**

```bash
tcpdump -t -n -vv -s 512 ip6 -i eth0
```

Appendix D

**Kernel settings in `/proc–filesystem`**

The virtual filesystem that we call `/proc` contains lots of different data structures and information gathered from the kernel at runtime, and updated whenever you try to list or view the information. However, most of the files available through the `/proc` filesystem are only available in read only mode, which means they can't be changed. This is because they only supply us with informational data. On the other hand, all of the variables located in `/proc/sys` (and the correspondent subdirectories) are writable as well as readable.

**How to set variables**

The `ipsysctl` variables may be set in two different ways which entails two totally different methods. The first one uses the `/proc` filesystem, which should come with any linux installation as long as you have a kernel that has `/proc` filesystem turned on. The other way is via the `sysctl` application provided with most distributions per default these days.

**Using `cat` and `echo` (/proc filesystem)**

Using `cat` and `echo` is the simplest way to access the `/proc` filesystem
You need to have read and sometimes also write access (normally root only) to the /proc–filesystem

- **Retrieving a value**
  The value of an entry can be retrieved using `cat`:
  ```
  cat /proc/sys/net/ipv6/conf/all/forwarding
  0
  ```

- **Setting a value**
  A new value can be set (if entry is writable) using "echo":
  ```
  echo "1">/proc/sys/net/ipv6/conf/all/forwarding
  ```

**Using sysctl**

Using the `sysctl` program to access the kernel switches is a common method today.

- **Retrieving a value**
  The value of an entry can be retrieved through:
  ```
  sysctl net.ipv6.conf.all.forwarding
  net.ipv6.conf.all.forwarding = 0
  ```

- **Setting a value**
  A new value can be set (if entry is writable):
  ```
  # sysctl -w net.ipv6.conf.all.forwarding=1
  net.ipv6.conf.all.forwarding = 1
  ```
  Note: Don't use spaces around the "=" on setting values. Also on multiple values per line, quote them like e.g.
  ```
  # sysctl -w net.ipv4.ip_local_port_range="32768 61000"
  net.ipv4.ip_local_port_range = 32768 61000
  ```

**Values**

There are several formats seen in /proc–filesystem:

- **BOOLEAN**: simple a "0" (false) or a "1" (true)

- **INTEGER**: an integer value can be unsigned, too more sophisticated lines with several values: sometimes a header line is displayed also, if not, have a look into the kernel source to retrieve information about the meaning of each value...

In /proc/sys/net/ipv6/... you can find plenty of IPv6 kernel parameters that can be configured at runtime.

Next you can find a small list of IPv6 related variables (Consult Documentation/ip-sysctl.txt to see all the existent variables)

- **use_tempaddr** - **INTEGER**
  Preference for Privacy Extensions (RFC3041).
  ```
  <= 0 : disable Privacy Extensions
  == 1 : enable Privacy Extensions, but prefer public addresses over temporary addresses.
  > 1 : enable Privacy Extensions and prefer temporary addresses over public addresses.
  ```
**Default**: 0 (for most devices)
-1 (for point-to-point devices and loopback devices)

- **dad_transmits** - INTEGER
  The amount of Duplicate Address Detection probes to send.
  Default: 1

- **mtu** - INTEGER
  Default Maximum Transfer Unit
  Default: 1280 (IPv6 required minimum)

- **router_solicitation_delay** - INTEGER
  Number of seconds to wait after interface is brought up before sending Router Solicitations.
  Default: 1

- **router_solicitation_interval** - INTEGER
  Number of seconds to wait between Router Solicitations.
  Default: 4

- **router_solicitations** - INTEGER
  Number of Router Solicitations to send until assuming no routers are present.
  Default: 3