Lab Notes

1 Setup

1.1 Caveats

- The bandwidth is shared by all of you: minimize bandwidth hungry activities.
- You're 20+ years old: even if the temptation is high, don't intentionally mess up the network.
- The computer has no brain. Use yours.
- Never copy and paste the tutorial commands, your brain will fix the concepts better if you copy them down by hand.
- If you feel frustrated when "the damn thing does not work", don't stop trying, it will work, eventually.
- The lab asks you to send confirmation messages to a specific IP after you've completed a portion of it. These messages will not be logged and used for grading, so there's no use in cheating.

1.2 Tools

- You should have a working linux distribution on your laptop, if this is not the case, get one *now* and install it. A virtual machine installation is fine, but it means extra-difficulties with the network configuration usually.
- You should have installed the following tools : iproute2 tcpdump nc ping wireshark tr. If you are missing any of them install them before starting the lab

2 Lab Goals

2.1 Basic Goals

- Connect to one of access points with ESSID: *group-N*, password (WPA2-PSK): *group-Nletmein*. If it is available, prefer the access point with the number given by your ID modulo 4.
- To obtain your ip address, split your ID, call a the last two digits, and b the 3rd and 4th. Your IP address is 10.2.(a+1).(b+1)/16.
- Configure properly your IP address, try and reach for 10.2.1.150 via ping (send one ping and stop).

- To check that the whole thing is working, connect via netcat to 10.2.1.150 on port 12345, you should receive a "Network Ok" string. Answer with your ID, followed by an enter. Disconnect with CTRL-C.

2.2 Further activities

- Network discovery: employ the `ping` command to send **ONE** broadcast ping to the whole 10.2.0.0/16 to know who's online. **One** ping means ONE, do not leave ping hammering the network. Redirect the output to a text file for easier consultation.

- If you prefer, you can perform network discovery with nmap. In this case employ timing templates slower or equal to the *polite* setting (i.e. `-T 2`)

- Open up two terminal emulators, place them side by side. On the first terminal emulator start a listening netcat on port 16000. On the second connect to the next (in PoliMi ID order) alive client looking it up in the text file you produced before. Employ the same 16000 port connect. Enjoy the chat.

- Connect via netcat to 10.2.1.150 on port 12346, you should receive a "Done chatting?" string. Answer with your ID, followed by an enter. Disconnect with CTRL-C.

2.3 Trickier activities

- Daisy chain: exploiting the two chats you have successfully brought up, now act as a relay between the two people you were chatting with: Set up a relay with two netcat instances daisy chaining the two people: this can be simply done through piping the two command together in bash. Check with them that they're able to communicate (i.e. chat) with each other through the relay you provide. Employ port 16001 and 16002 for the relay.

- Set up a flame quenching relay for chats: Sometimes, the mood in a chat starts to be aggressive, and people start shouting in CAPITALS. Since this is annoying, employ the `tr` tool to suppress all capital letters which are transiting through your relay. `tr` by default acts on stdin and outputs the result on stdout, so it's easy to plug it into the previous relay.

- Compress the communications: employ gzip to compress your communications via netcat. gzip can be employed to compress also input coming from stdin, by simply invoking it with no input file. The output of the command can be redirected to stdout with the option `-c`. Try sending a small text file via netcat and verify through packet inspection with wireshark that it has effectively been compressed.

- Once you've got here, Connect via netcat to 10.2.1.150 on port 12347, you should receive a "I see 0xDEAD packets." string. Answer with your ID, followed by an enter. Disconnect with CTRL-C. Point out to the teacher that you got here.

- Hunt server: download the package bsdgames and contact your two chat mates, pull up a hunt server and start playing. Examine the traffic and how hunt handles the connections.

A Appendix - Small tutorial for connection to WPA2-PSK networks

Connecting to a protected wireless network requires a level 2 setup which is slightly different from the common `ip link set wlan0 up`. Namely, you need to:

- Enable the interface via `ip link set yourwirelesscard up`
- You will need to lock on a specific access point, this can be achieved via `iwconfig yourwirelesscard essid accesspointname`
- In case the access point does not require any authentication, you're set. However, since we will be using an encrypted network, you'll also need to use $\texttt{wpa}_s upplicant to connect. \texttt{wpa}_s upplicant needs a small configuration file to know to whom it should be aut$
- ```
  network={
      ssid="essid"
      psk="password"
      proto=RSN
      key_mgmt=WPA-PSK
      pairwise=CCMP TKIP
      group=CCMP TKIP
      }
  ```

To run $\texttt{wpa}_s upplicant you will need to supply the following command line parameters$:
$\texttt{Ddriver}_n ame$: $usually the Linux wireless extensions \texttt{wext} work fine \texttt{-i interface}$: $check the name of your wireless card, it usually is wlan0$

$\texttt{-c configuration}_f ile$: $ok, this one is pretty self - explanatory.$
  ```