Kernel Training Final Assignment

Overview

You will have to demonstrate your knowledge in several areas that were covered during the training such as **`file operations`**, **`networking`**, and **`timers`**. Bonus points will be granted for demonstration of self-education skills.

The base code for the assignment is the solution that was presented in class for the ARP_IN netfiler exercise (**net_filt.c**).

Details

- The original code prints the **`MAC`** address of the source of inbound **`ARP`** requests.
- Store the **`MAC`** addresses in a LIFO style cache, instead of printing them:
  - The cache should store up to 50 **non-unique** addresses.
  - Each entry should store the source MAC address, source IP and the name of the network device via which the request arrived.
  - Entries should have an expiration time. The expiration time will be tunable via a module parameter.
  - Bonus: Make the parameter mutable after module load time.
  - Bonus: Implement that **`MAC`** address cache using the kernel built-in **`list`** infrastructure.
- Implement a pseudo file under **`/proc`**:
  - Read requests should return the content of **`MAC`** address cache in a human readable form.
  - Strings written to the proc entry should be treated as numbers (ignore non numeric strings). The number written will be a count of how many entries to immediately remove from the **`MAC`** address cache. The entire cache should be cleared, if the number exceeds the current count of cache entries.

Deliverables

- The module's source code and **`Makefile`**.
- A **`readme`** file, that will detail how to use the module, and will provide a brief documentation of haw was the module implemented.

Miscellaneous

- Due date – December 12, 2021
- Do you have any question? Send an email to allon@thought.co.il (replies will be given by the end of the following business day).

Appendix

- Kernel built-in **`list`** API is defined in **`linux/list.h`**.
- Use the **`name`** field of the **`net_device`** structure to get the device's name.
- The offset of the IP header may be retrieved via **`SKB_NETWORK_HEADER (sk_buff)`**.
- IP header layout: