TCP SACK

The goal of this assignment is to print TCP SACK acknowledgements in the TCP optional header. You are to implement the `print_tcp_options` routine that takes the starting address of the optional header and the length of the optional header as input.

The option header contains values of different kinds. There could be multiple values of different kinds stored in the optional header. The starting byte stores a *kind*.

If the kind is zero, it means the end of options has been reached, and there is no need to look at the rest of the header.

If the kind is one, it means it is a nop (no-operation), and you can skip this byte. The following byte stores a kind.

If the kind > 1, the following byte contains the length. After the length, a value is stored. The length (in bytes) includes the size of the value + 2 (i.e., bytes corresponding to kind and length).

You can skip the value to find the next kind. Be careful before reading the next byte; it must be within the length of the optional header.

If kind = 5, the value is a sequence of ranges received by the receiver beyond the cumulative acknowledgement number. Each range is eight bytes long, as the sequence number is four bytes long. The number of ranges can be computed using (len - 2)/8, where len is the value stored after kind (5).

To summarize, a TCP optional header looks like:
kind | len | value | kind | len | value | kind | len | value | ....

The kind and len fields are of one byte. If kind is one, the subsequent byte is a kind, i.e., no len or value field. If you encounter kind = 0, stop parsing; otherwise, parse until you reach the end of the header. If the TCP optional header contains kind = 5, you need to print SACK ranges; otherwise, don't print anything.

Skeleton code:

Clone the git repository using
`git clone https://github.com/Systems-IIITD/SACK`
Run **`make`** to compile pcap.c.
Run **`sudo ./pcap`** to run the tool.

The `pcap` application uses the `pcap` library to read all packets from the network interface card. It then filters the TCP packets with optional headers and calls print_tcp_options, which

you are to implement.

Experiments:

You will need two machines for this experiment.
Check if TCP SACK is enabled using the following command.
`sysctl net.ipv4.tcp_sack`
It should print: **`net.ipv4.tcp_sack = 1`** if SACK is enabled.

Otherwise, enable SACK using:
`sudo sysctl -w net.ipv4.tcp_sack=1`
Repeat the same steps on the other system.

Force one of the systems to drop 10% packets using:
`Sudo tc qdisc add dev eth0 root netem loss 10%`
Replace `eth0` with the name of your network interface.

You can restore the original behaviour (i.e., no packet drop) using:
`sudo tc qdisc del dev eth0 root` after this experiment.

The packet drop will increase the chances of SACK acknowledgements.
Run **`sudo ./pcap`** on your system after implementing your logic.

Transfer a large file (greater than 1 GB) from the machine on which you are running `pcap`. You can use `scp` to transfer the file. You should see SACK acknowledgements in the `pcap` window. Collect at least 100 log messages for submission.

Transfer the file with and without TCP SACK. You can disable TCP SACK using:
`sudo sysctl -w net.ipv4.tcp_sack=0`

Submission:

You need to submit a design documentation along with the pcap.c file and an output log file containing the SACK ACKs printed by your implementation (at least 100). Use the submission guidelines for the homework and assignment.

Answer the following questions in the design documentation.

1. How much time does it take to transfer a 1 GB file with TCP SACK enabled, and you drop 10% of the packets at the receiver? Use /usr/bin/time scp … to get the total time. 2. How much time does it take to transfer a 1 GB file with TCP SACK disabled, and you drop 10% of the packets at the receiver?
3. What is the total number of TCP SACK ACKs received during the transfer of the file, with SACK enabled?
4. Out of all the TCP SACK received, how many times were the numbers of ranges 1, 2, and 3?

5. What are the other kinds you have encountered apart from 0, 1, and 5? Briefly describe their purpose.