Advanced Networking
Università della Svizzera italiana — A. Carzaniga, P. Polverino

Assignment n. 1 (graded)
March 14, 2024

Assignment 1: Exploring Congestion Control

Due date: *Sunday, March 24, 2024 at 22:00*

This is an individual assignment. You may discuss it with others, but your code and documentation must be written on your own.

Part I: Socket Programming

In a source file called `client.c` write a C program that connects to a TCP server and sends some pseudo-random generated data. The general usage of the `client` program is as follows:

`client` [*options*] *server*

This means that the program takes zero or more *options* and the IP address or the hostname of the server. The following *help* page describes the *options* your implementation should support.

```
$ ./client --help
Usage: client [options] <server>

-p <port>, --port <port>
                  The port the server is listening on (default: 5000)
-n <size>, --size <size>
                  The number of pseudo-random bytes to send to the server (default: 1000)
-c <congestion>, --congestion <congestion>
                  The TCP congestion control algorithm to use (default: cubic)
-h, --help
                  Display this help and exit
```

You can download the server program from the iCorsi system. Note that setting the congestion control algorithm on macOS might not be possible, but you could use a VM to test your implementation.

Part II: Congestion Control Algorithm Comparison

Use the `client` and `server` programs to generate some traffic with different congestion control algorithms on the client. Then, try to estimate the congestion window (`cwnd`) for the different captures and analyze how it changes based on the congestion control algorithm used. Some examples of algorithms you might want to experiment with are the Reno, Cubic, and Vegas algorithms. You may use the `rate-limiter.sh` script from the iCorsi system to limit the transmission rate. On a Linux machine, you can use the following commands to check which congestion control algorithms are available on your system

```
$ ls -l /lib/modules/$(uname -r)/kernel/net/ipv4
```

To check which congestion control algorithms are currently allowed on your machine, you can use the following command

```
$ sysctl net.ipv4.tcp_allowed_congestion_control
```

To make a congestion control algorithm available, you need to load it as a kernel module with the following command

```
$ sudo modprobe -a tcp_<algorithm>
```

Finally, you can allow the congestion control algorithm just loaded with the following command

```
$ sudo sysctl -w net.ipv4.tcp_allowed_congestion_control='<list of algorithms>'
```

Submission Instructions

Submit a source file named `client.c` and a brief report about your experiments on the different congestion control algorithms, through the iCorsi system. Add comments to your code to explain sections of the code that might not be clear. You must also add comments at the beginning of the source file to properly acknowledge any and all external sources of information you may have used, including code, suggestions, and comments from other students. If your implementation has limitations and errors you are aware of (and were unable to fix), then list those as well in the initial comments.

You may use an integrated development environment (IDE) of your choice. However, *do not submit any IDE-specific file*, such as project description files.