CSCE 464 Wireless and Mobile Systems

Homework 2

(285 points; Submit through canvas.tamu.edu)

Name: Your Name

You MUST use this latex template for writing your answers to questions that require written answers. You are free to use any Latex editor you wish. A free, friendly version is at overleaf.com. Upload the two latex files provided and you will be ready in a few minutes to edit them and produce a pdf.

1 Multipath TCP (MPTCP)

1.1 Hands-on MPTCP in Ubuntu 22.04 LTS [120 points]

In this part of the assignment you will gain hands-on experience with MPTCP and tcpdump (command line packet analyzer, or a version of Wireshark for environments where Wireshark GUI is not possible).

MPTCP is an integral part of 5G [1]. There have been ongoing efforts to upstream MPTCP into the Linux kernel [2]. In this part of the assignment, we will enable MPTCP support in Ubuntu 22.04 LTS and monitor an MPTCP connection to understand its functionality.

1.1.1 Become familiar with tcpdump

Check the man pages for tcpdump (# man tcpdump). Here are a few examples on how to use it:

```
$sudo tcpdump -i eth0 -v port 1234
```

which monitors interface eth0 and port 1234 and prints output in verbose mode

```
$sudo tcpdump -i eth0 udp
```

which monitors interface eth0 and filters out other packets than udp

```
$sudo tcpdump -i eth0 dst 10.10.1.20
```

which monitors interface eth0 and filters out other packets than those with destination 10.10.1.20. Alternatively, you can use ‘src’ in place of ‘dst’.

1.2 What To Do

1.2.1 Enable MPTCP and set up network interfaces

In VirtualBox setup 2 Virtual machines to act as the server and the client. You can set up the second VM in the same way as the previous one (in HW1). Configure 2 network interfaces in the server VM and 1 interface in the client VM. Set all network interfaces in VirtualBox as Bridged Adapter.

Enable MPTCP in both VMs as follows:

```
$sudo sysctl net.mptcp.enabled = 1
```

1.3 Setup signal and subflows on the server and client

Follow the given steps below in the server VM to setup a signal and subflow. Use the ifconfig command to identify the IP addresses of the machines in the Ethernet interface.

Server side:

1. Set the subflow limit to 1

```
$sudo ip mptcp limits set subflow 1
```

2. Set the number of addresses accepted to 1

```
$sudo ip mptcp limits set subflow 1 add_addr_accepted 1
```

3. Add one of the interface's IP address as a signal and the other one as a subflow

```
$sudo ip mptcp endpoint add <one of the server interface's IP>
dev <corresponding interface name> signal
$sudo ip mptcp endpoint add <server's other interface IP>
dev <corresponding interface name> subflow
```

Client side:

1. Repeat step 2 from the server side.

1.3.1 Capture Traffic from a Sample Application

Study the provided code that creates a server and multiple clients communicating using MPTCP. Compile and execute the code.

Server:

```
./server <port number> files/small.txt <number of times the message is to be sent>
```

Client:

```
./client <Server's subflow interface IP> <port number>
```

Before you start to capture traffic ensure that the signal and subflows are properly set. View and confirm this using the following command:

```
$sudo ip mptcp endpoint show
```

Capture the traffic using tcpdump and check for MPTCP. Try to also capture the traffic and observe subflows using

```
$sudo ip mptcp monitor
```

Take screenshots of both. Attach them below in Figure 1 and 2 [Please make sure your figure is legible. Increase the size of the figure, if needed.

Figure 1: MPTCP for a sample application - tcpdump

Figure 2: MPTCP for a sample application - monitor

2 SSL/TLS Client Server Application (45pts)

In this part of the assignment you will learn how to use certificates and secure communication between a client and a server using TLS/SSL.

Start by studying, compiling and executing the attached sample client and server (in tls-ssl-client-server.tar). The make_cert.sh script contains the command for creating a public-private key pair (cert.pem and key.pem).

To compile and execute the client and the server, execute:

```
$gcc -o client client.c -lssl -lcrypto
$gcc -o server server.c -lssl -lcrypto
$./server 1234 &
... you will be prompted for the password to the pem file, which is: "password"
$./client localhost 1234
```

2.1 What To Do

Modify your submitted pincracktest server and client to use TLS/SSL for communication.

2.2 What to Submit

Submit the modified pincracktest client and server that use TLS/SSL. Submit a screenshot of the Wireshark capture for the communication between the client and the server with AND without TLS/SSL as shown in Figure 3.

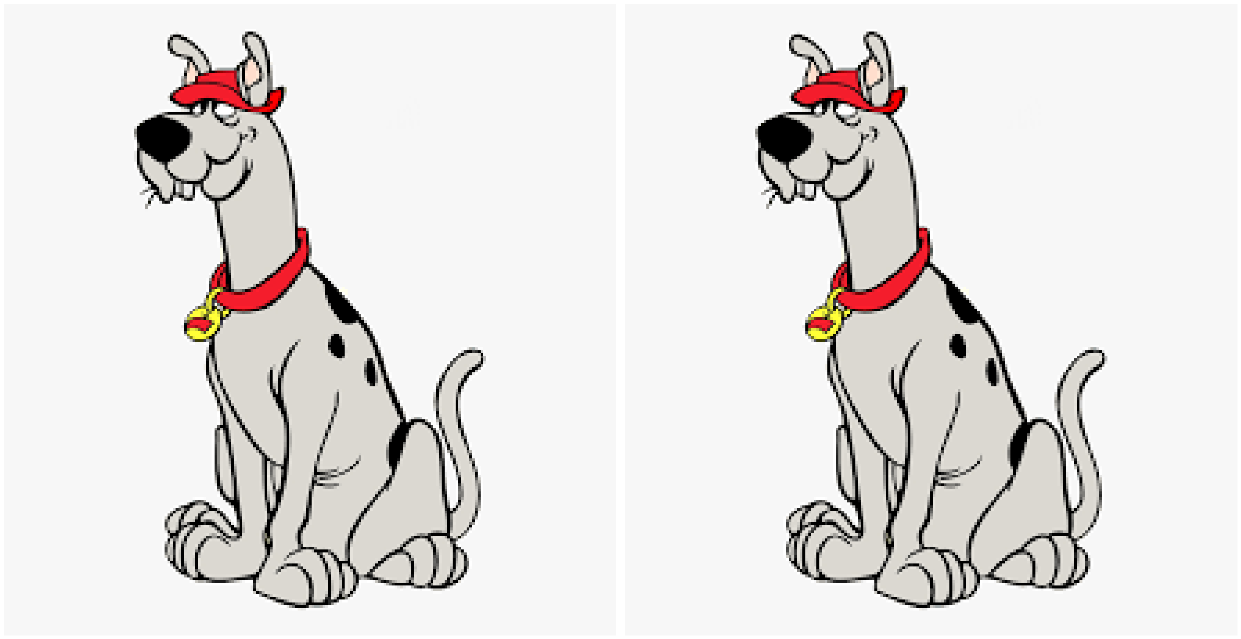

Figure 3: Wireshark output without (left) and with (right) TLS/SSL

3 A QUIC Library (msquic) and Sample Application (60pts)

In this part of the assignment you will experiment with an implementaiton of th QUIC protocol, as provided by Microsoft [3]. We will largely follow the build steps suggested and you will experiment with a client and server applications that use QUIC (libmsquic.so).

3.1 Build Instructions for msquic

These instructions are for a Linux Ubuntu platform. Microsoft gives instructions on how to build msquic on Windows and Mac. However, for this assignment you will be on your own if you work on Windows or Mac.

First, you need to clone the git repository, by executing the following:

```
$git clone https://github.com/microsoft/msquic.git
$cd msquic
$git submodule update --init --recursive
```

In particular, the last command is important as some subprojects in msquic are external modules, and they need to be fetched recursively, separately.

Next, edit the file ~/msquic/CMakeLists.txt and change:
option(QUIC_BUILD_TOOLS "Builds the tools code" OFF)
to:
option(QUIC_BUILD_TOOLS "Builds the tools code" ON)

Now you are ready to build the library:

```
$mkdir build && cd build
$cmake -G 'Unix Makefiles' ..
$cmake --build .
```

If you encounter problems during the build, likely your system is missing some dependencies, e.g., cmake, or openssl, etc. To ensure you have all dependencies satisfied, check the "Documentation" [3] and its "Build docs" section.

3.2 A quicsample Application showcasing the QUIC Protocol

The msquic implementation of the QUIC protocol provides several sample applications that showcase the use of QUIC. The one we will use in this assignment is the following:

```
$~/msquic/src/tools/sample/sample.c
```

which contains both client and server code. Study the source code, how the client and the server need to be executed (i.e., command line arguments). Study also the source code for instructions on how to create the certificates that the server will need, in order to start. Create those certificate as instructed in the source code.

The executable for the sample application is built by cmake and it is located in:

```
$~/msquic/build/bin/Release/quicsample
```

Start the client and the server. As mentioned above, the server will need two certificates: a public key (:cert_file) and a private key (:key_file).

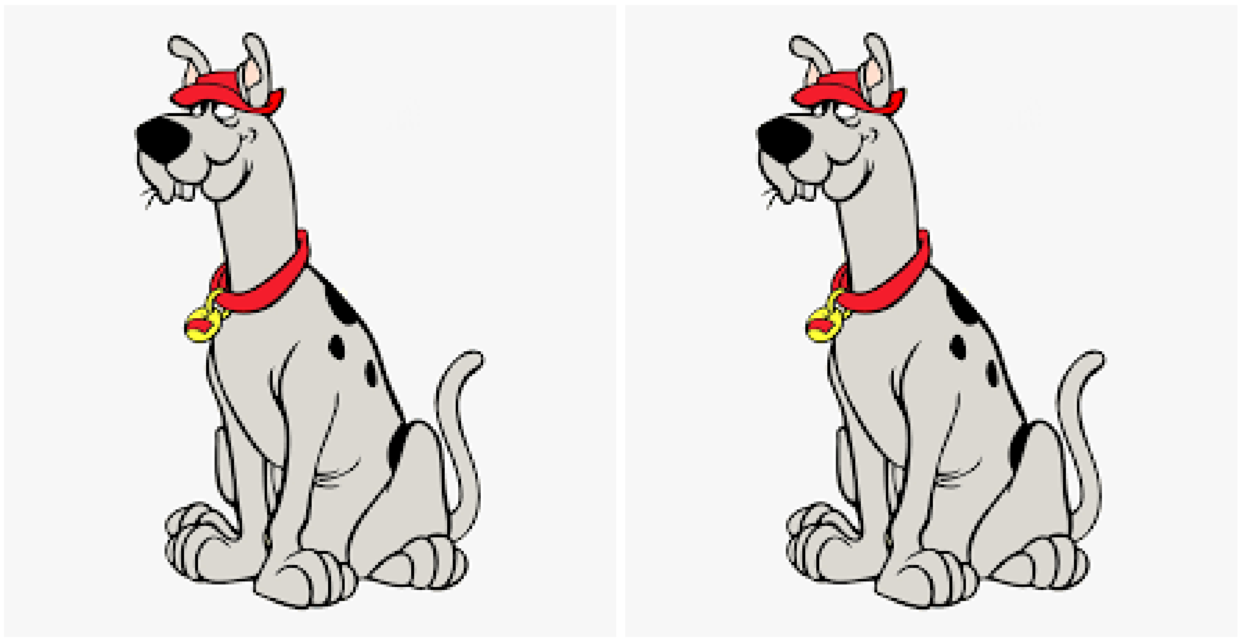

Figure 4: Client (left) and Server (right) output

3.3 What to Submit

Attach two screenshots showing the output from the server and from the client once a connection is established (Figure 4). [Please make sure your figure is legible once the pdf is created. Increase figure size accordingly]

Attach one screenshots showing the output Wireshark once the client communicated successfully with the server (Figure 5). [Please make sure your figure is legible once the pdf is created. Increase figure size accordingly]

4 MPBond (60pts)

Please answer the following questions.

1. (2 points) What is the problem that MPBond tried to solve?
   Answer: Write your answer here

2. (2 points) What is/are the main key idea(s) for solving the problem?
   Answer: Write your answer here

3. (2 points) What applications can potentially benefit from the MPBond solution?
   Answer: Write your answer here

4. (2 points) Explain the "simultaneous subflow completion" problem that is present in MPTCP and MPBond tries to solve? Why does it occur in MPTCP?
   Answer: Write your answer here

Figure 5: Wireshark output

5. (2 points) Explaing the Dual Mode in MPBond?
   Answer: Write your answer here

6. (20 points) Describe the kibbutz solution that MPBond compares against (short paragraph) Answer:
   Answer: Write your answer here

7. (20 points) Describe the COMBINE solution that MPBond compares against (short paragraph) Answer:
   Answer: Write your answer here

8. (10 points) Describe the condition that triggers a reinjection of a packet in MPBond and the rationale behind it. (short paragraph) Answer:
   Answer: Write your answer here

5 Submission Instructions

Please submit the PDF as obtained from the provided latex template. Name your submission as ⟨your lastname⟩.hw2.pdf and submit it on canvas.tamu.edu. Also, please submit all source code as a single zip or tar file.

References

[1] Lan Ding and Ye Tian and Tong Liu and Zhongxiang Wei and Xinming Zhang, Understanding commercial 5G and its implications to (Multipath) TCP

[2] Linux kernel discussion on MPTCP (Sept 2021): `https://lore.kernel.org/mptcp/`

[3] Microsoft msquic, A General Purpose QUIC library: `https://github.com/microsoft/msquic`